UNIVERSIDADE FEDERAL DO ESTADO DO RIO DE JANEIRO
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

LEARNING DATA-AWARE DECLARATIVE MODELS FROM PROVENANCE DATA

Mateus Ferreira Silva

Orientadoras
Prof. Dra. Fernanda Araujo Baião Amorim
Prof. Dra. Kate Cerqueira Revoredo

RIO DE JANEIRO, RJ – BRASIL
Setembro de 2015

LEARNING DATA-AWARE DECLARATIVE MODELS
FROM PROVENANCE DATA

Mateus Ferreira Silva

DISSERTAÇÃO APRESENTADA COMO REQUISITO PARCIAL PARA OBTENÇÃO DO TÍTULO DE MESTRE PELO PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA DA UNIVERSIDADE FEDERAL DO ESTADO DO RIO DE JANEIRO (UNIRIO). APROVADA PELA COMISSÃO EXAMINADORA ABAIXO ASSINADA.

Aprovada por:

[illegible]

Prof. Daniel Cardoso Moraes de Oliveira, D.Sc. – UFF

[illegible]

Prof. Flávia Maria Santoro, D.Sc. – UNIRIO

[illegible]

Prof. Fernanda Araujo Baião Amorim, D.Sc. – UNIRIO

[illegible]

Prof. Kate Cerqueira Revoredo, D.Sc. – UNIRIO

RIO DE JANEIRO, RJ – BRASIL
Setembro de 2015

Silva, Mateus Ferreira.

S586 Learning data-aware declarative models from provenance data / Mateus Ferreira Silva, 2015.

62 f. ; 30 cm

Orientadora: Fernanda Araujo Baião Amorim.

Coorientadora: Kate Cerqueira Revoredo.

Dissertação (Mestrado em Informática) - Universidade Federal do Estado do Rio de Janeiro, Rio de Janeiro, 2015.

1. Data-aware. 2. Metadados. 3. Modelo declarativo. 4. Dados de proveniência. 5. Workflow científico. I. Amorim, Fernanda Araujo Baião. II. Revoredo, Kate Cerqueira. III. Universidade Federal do Estado do Rio de Janeiro. Centro de Ciências Exatas e Tecnológicas. Curso de Mestrado em Informática. IV. Título.

CDD – 025.04

Para minha filha Laura que enche nossos corações de amor.

Agradecimentos

Ser aluno de mestrado é uma tarefa que necessita de apoio familiar, compreensão dos amigos e de colegas trabalho, ajuda dos contemporâneos do mestrado, orientação de professores e um auxílio divino para ter persistência e inspiração. Por ter tido tudo isso, agradeço:

A Deus que em sua infinita sabedoria tem me dado força e sabedoria para atingir as metas que traço para minha vida.

A minha esposa Carolina que sempre acreditou em meu potencial, pela paciência nos momentos de estudo, incentivando-me nos momentos de dúvida. E a nossa filha Laura, nosso maior bem, que foi muito desejada por todo período de mestrado e está perto de nascer, nos trazendo muita alegria nesse último ano de mestrado.

Aos meus pais, Adeládio e Claudemira, por tudo que fizeram pela minha educação. A minhas irmãs Tatiana e Tâmara e ao meu irmão Marcelo, que sempre foram fonte de incentivo para alcançar meus objetivos.

Aos professores do Departamento de Informática Aplicada da UNIRIO, com eles pude aumentar meu conhecimento na ciência da computação. E em especial, as minhas orientadoras Fernanda e Kate que propiciaram uma excelente orientação, pelas boas discussões durante a pesquisa, pelo conhecimento compartilhado comigo e pela paciência ao lidar com minhas dificuldades, não me deixando sair da trilha. A presença delas está obviamente manifestada em cada linha deste trabalho. Posso dizer, com certeza, que o convívio com elas durante o mestrado aumentou minha admiração pelo profissionalismo e pelo notório saber que elas possuem.

Aos colegas da UNIRIO João Carlos Gonçalves, Fernando Xavier, Pedro Richetti e Felipe Leão, pelos conhecimentos compartilhados que contribuíram com abordagem dessa pesquisa.

E aos colegas da Petrobras: Jeferson Rocha, Margareth Cruz, Manoel Alexandre Ricardo Monteiro, Ricardo Neimo e Vinícius Ribas pela compreensão nas minhas ausências e incentivo a continuidade do mestrado.

SILVA, Mateus Ferreira. **LEARNING DATA-AWARE DECLARATIVE MODELS FROM PROVENANCE DATA**. UNIRIO, 2015. 62 páginas. Dissertação de Mestrado. Departamento de Informática Aplicada, UNIRIO.

ABSTRACT

Data provenance represents a collection of metadata on the origin and history of data. In scientific workflows, this metadata is essential for operational support of scientific experiments. The amount of provenance data generated from scientific workflow executions grows exponentially through time, becoming infeasible for scientists to manually analyze its content. Thus, mechanisms for extracting and modeling the knowledge implicit in provenance data are demanding. Due to the diversity and flexibility inherent to scientific experimentation scenarios, declarative models are potentially adequate for the task. However, they typically do not consider data attributes, which would enrich its embedded knowledge with relevant information such as parameter values used in each workflow instance. A classification model may fill this gap. This work proposes an approach to automatically learn both declarative and a classification models from provenance data, and combine them into a unique view. This proposed approach was evaluated on two real scientific experiments scenarios on the domain of text mining and of Evapotranspiration estimation.

Keywords: data-aware, declarative model, provenance data, scientific workflow

RESUMO

A proveniência de dados representa uma coleção de metadados sobre a origem e histórico dos dados. Em *workflows* científicos, este metadados é essencial para o apoio operacional de experimentos científicos. A quantidade de dados de proveniência gerados a partir de execuções de *workflows* científicos cresce exponencialmente ao longo do tempo, tornando-se inviável para os cientistas analisar manualmente o seu conteúdo. Assim, mecanismos para extrair e modelar o conhecimento implícito nos dados de proveniência são demandados. Devido à diversidade e flexibilidade inerente aos cenários de experimentação científica, modelos declarativos são potencialmente adequados para esta tarefa. Entretanto, eles tipicamente não consideram atributos de dados, o que poderia enriquecer seu conhecimento incorporado com informações relevantes, tais como os valores dos parâmetros utilizados em cada instância do *workflow*. Um modelo de classificação pode preencher esta lacuna. Este trabalho propõe uma abordagem para aprender automaticamente tanto modelos declarativos e de classificação a partir de dados de proveniência, e combiná-los em uma única visão. Esta abordagem proposta foi avaliada em dois cenários reais experimentos científicos no domínio de mineração de texto e de estimativa da evapotranspiração.

Palavras-chave: *data-aware*, modelo declarativo, dados de proveniência, *workflow* científico

Table of Contents

List of Figures

List of Tables

List of Abbreviations

BDMEP	Meteorological Database for Education and Research
BPM	Business Process Management
BPMN	Business Process Model and Notation
BPEL	Business Process Execution Language
CRG	Compliance Rule Graph
DCR	Dynamic Condition Response Graphs
INMET	National Institute of Meteorology of Brazil
LTL	Linear Temporal Logic
MTC	Many Tasks Computing
PROV-DM	PROV Data Model
RDBMS	Relational Database Management System
SWfMS	Scientific Workflow Management System
UML AD	Unified Modeling Language Activity Diagram

W3C	World Wide Web Consortium

Chapter 1 – Introduction

This chapter introduces the topics that motivated the research. The problem under analysis is characterized, the hypothesis is established, research goals defined and the method used explained.

1.1 Motivation and Problem Characterization

Scientific experimentation is a method applied in many fields of science, such as: physics, chemistry, biology, bioinformatics, astronomy, cosmology, meteorology, oceanography and agriculture (CUEVAS-VICENTTÍN, 2012; MEDEIROS, 2011).

Scientific experimentation is an interactive process, triggered by questions about an observed phenomenon, followed by hypothesis development and test executions using several variations of the studied scenario. The scientist fine-tunes the experiment by modifying tasks and parameters until their hypothesis has been accepted, refuted or modified, and process is finished (ZENG *et al.*, 2011). Scientific learning is an iterative process, which begins with the current scientific knowledge and then chooses a theory to test or explore (SELTMAN, 2015).

A Scientist or a research team may use a Scientific Workflow Management System (SWfMS) to support the execution of scientific experiments in data-flow scenarios modeled as workflows, varying input data, parameters, or algorithms. SWfMS (OLIVEIRA *et al.*, 2010). SWfMS supports scientist with construction of new experiments, re-execution, documentation, reuse and provenance management (MEDEIROS, 2011).

The historical data generated during the execution of scientific workflows is named retrospective provenance data. It encompasses information about activities, agents, execution timestamps, parameters, and entities (LIM *et al.*, 2010), so as to answer important questions regarding the experiment rationale, such as what activities were performed by an agent in a time period, what are the input parameters that led to the best results, or which data artifacts were derived. The varying range of each parameter is very important in some experiments, since the scientist can have different results depending on the chosen parameter, so, it is valuable information when analyzing workflow provenance data.

Additionally, prospective provenance data includes the abstract workflow specification for a particular scientific experiment, which may be specified, detailed, and then executed by means of a workflow engine (LIM *et al.*, 2010). In general, prospective provenance data is procedural and imperative; *i.e.*, requires the prediction of all possible paths (VAN DER AALST *et al.*, 2009).

Both retrospective and prospective views are important to manage, understand, and reproduce experiments (DEELMAN; CHERVENAK, 2008; LIM *et al.*, 2010) and therefore should be carefully and constantly analyzed by the scientist when evaluating past executions and planning future executions.

However, the amount of provenance data generated from scientific workflow executions grows exponentially through time, becoming infeasible for scientists to manually analyze or evaluate its content. Moreover, it is essential to identify useful information (CUEVAS-VICENTTÍN *et al.* 2012) based on user-requirements. Therefore, automatic mechanisms for provenance data analysis are demanding.

On the other hand, Process Mining is an advanced technique to automatically discover process models from data logs (MAGGI et al., 2011). As such, it may be considered a powerful technique for provenance data analysis in scientific experiment scenarios, where the data log is constituted by the set of workflow instances records. Typically, the data log is configured by the scientist to register relevant provenance data, as well as the results reflecting how well the result of each workflow instance achieved the experiment objectives (VAN DER AALST *et al.* 2012; TERUEL *et al.*, 2014). In our work, the experiment objectives and logged relevant provenance data constitutes user-defined requirements.

There is a plethora of process modeling languages, some of them used to explicit a procedural view of the process in a imperative model (such as BPMN, Epics, Petri nets, BPEL, UML activity diagrams (VAN DER AALST *et al*, 2012)), and others able to explicit a declarative perspective of it [such as DCR graphs and Declare (MAGGI et *al.*, 2011)]. Although it is possible to use both perspectives when representing a workflow (PESIC *et al.*, 2010), declarative models are more flexible than imperative models and are thus recognized as being more adequate to explicit valuable knowledge that may be embedded within scientific experiment provenance data (VAN DER AALST *et al.*, 2009). Flexible and procedural process model are not exclude each other. It is possible use different paradigms in order to provide flexibility in workflow (PESIC *et al.*, 2010).

However, declarative models are not typically designed to capture data aspects of a process (KNUPLESCH *et al.* 2013; MAGGI *et al.* 2013), which is of specific importance to capture implicit knowledge about parameters variation during a scientific experiment. This issue is reinforced by the Process Mining Manifesto, which claims that combining process mining with other types of analysis is a challenge (VAN DER AALST *et al.* 2012). A data-aware declare model can be more accurate, therefore, providing more reliable information to users.

Given the facts stated above, the problem under analysis in this work can be defined as: **How to learn a model that combines data-aware aspects with declare constraints based on provenance data?**

1.2 Hypothesis and Solution Proposal

The hypothesis guiding this research is stated as: **IF** declarative constraints and attribute data rules are combined, **THEN** more accurate models are discovered from provenance data.

We address the given problem by proposing an approach that captures both data aspects and workflow rules of a scientific experiment by applying process mining techniques combined with classification techniques on top of its provenance data. The approach learns a model that comprehends and inter-relates declarative rules and data-aware rules learned through declare miner and decision tree algorithms, respectively. Provenance data used for mining is based on user predefined requirements, *i.e.*, metrics that will provide information about whether each instance complies or not to the user quality requirements. The resulting combined model, a data-aware declarative model, provides operational support to have more accurate information about the experiments, *e.g.*, a valuable insight for scientists in understanding the main characteristics of their experiments that led to successful and unsuccessful workflow instances. A successful workflow instance is the one that finished its execution with no errors, and which results met all user-defined quality requirements. This approach supports scientists to plan future executions or performing conformance checks of semantic constraints in workflow activity.

1.3 Research Goal

The contribution of this dissertation is to present an approach to learn and represent a data-aware declarative model in the context of scientific experiments. We also provide the formal semantics of data-aware declare constraints, extended from the original declare templates (MAGGI *et al.*, 2011).

1.4 Research Methodology

The scientific methodology we followed in this research consisted of the following steps: first, a bibliographic review was conducted aiming at gathering information about recent generation of models from provenance data. After that, the motivation and problem were defined, with a hypothesis and set of goals being formulated to guide the research. The third step comprised the proposal specification, architectural design [in which a relational schema for the provenance dataset was designed as an extension to the PROV-DM metamodel (MOREAU; MISSIER, 2013), and implementation. The proposal was evaluated twofold. First, an exploratory case study was conducted using a data mining experiment as a scenario. This study attested the feasibility of our proposal, the correctness of our implemented architecture and gathered preliminary results from the chosen data- and process- mining algorithms. In addition, an experiment was conducted in the domain of evapotranspiration estimation. In this second scenario, the resulting data-aware declarative model was quantitatively analyzed using compliance verification metric against the original process model, to verify the research hypothesis.

1.5 Document Structure

This dissertation is structured into 6 chapters besides this Introduction. Chapters 2 and 3 present important concepts for understanding our approach, which is presented in Chapter 4 and evaluated in Chapter 5. Related Works are presented in Chapter 6. Finally, Chapter 7 concludes the paper and provides some future directions.

Chapter 2 – Scientific Workflows

A scientific experiment may be defined as a set of controlled actions that includes variations of tests, and whose results are usually compared to each other to accept or refute a hypothesis. This set of actions can be modeled as a scientific workflow, and each different workflow execution (trial) is named a workflow instance (MATTOSO *et al.*, 2010). Scientific workflows aim to accelerate scientific advances through task automation, scaling, simulation, etc (CUEVAS-VICENTTÍN et al. 2012).

The execution of one workflow is a part of life cycle of the scientific experiment by a SWfMS (MATTOSO *et al.*, 2010). Beyond the execution phase, there are more three phases: composition, configuration and analysis (Figure 1). Scientists draw up specifications in the composition phase of the experiment, stating which programs should be implemented and which are data dependencies between them. The next step is the configuration phase; in this moment computing resources are mapped for performing data transfer and processing; and workflow monitoring. Next, in analysis phase the data are available for the scientist check the experiments results. Scientists will draw conclusions based on query, visualizing and analyzing the data (OLIVEIRA, 2012).

Scientists can execute different programs in a flow of activities. These activities are performed in chain and they produce data as input to other activity. Furthermore, variations occur input data and parameters, like when then experiment demanded parameter sweep or loading different datasets. In other words, a scientific workflow W is represented by four elements (A, Pt, I, O), where: A is a chain of activities {a_1,a_2,a_3, …, a_n}; Pt is a set parameters of A {pt_1, pt_2, pt_3, …, pt_m}; I is a set of data inputs {i_1,i_2, i_3, …, i_r}; and O is a set of output data {o_1, o_2, o_3, o_s}(OLIVEIRA *et al.*, 2010). For example, a data mining experiment is modeled such a scientific workflow W. W has activities {a_1, a_2, a_3, a_4} where a_1=”data loader”, a_2=”data pre-processing”, a_3=”algorithm execution” and a_4=”data pos-processing”. These activities are performed against in different inputs $\{i_1, i_2, i_3, \ldots, i_s\}$ and parameter values $\{pt_1, pt_2, pt_3, \ldots, pt_m\}$ up to the experiment exploration be finished and the output data O achieves the expected result (OLIVEIRA *et al.*, 2010).

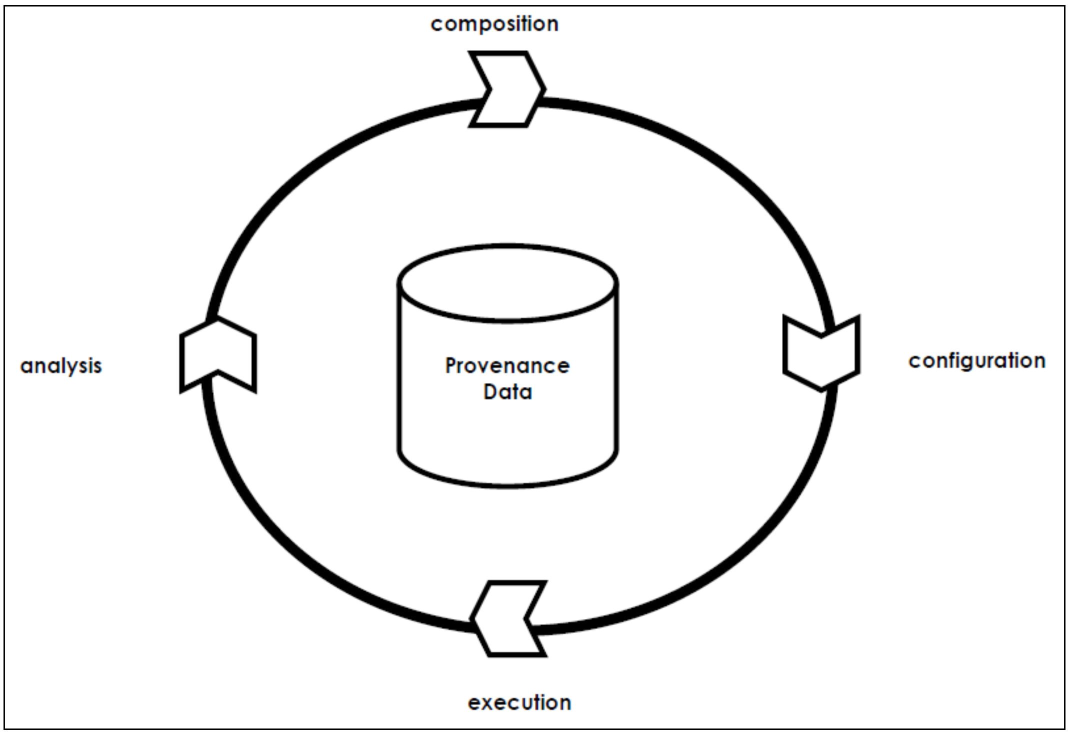


Figure 1 – Scientific Experiment Life Cycle (Adapted from OLIVEIRA, 2012)

Many real-world scenarios may be expressed as workflows. Complex workflows require a level of abstraction to help scientists express their experiments. This abstraction is offered by Scientific Workflow Management System (SWfMS) that model, execute, and monitor workflows (OLIVEIRA *et al.*, 2010). SWfMS supports scientists in planning different scenarios for experimentation, considering variations on input data, parameters, and algorithms. They can offer: run-time tasks management, resource capabilities, task scheduling, data provenance management, data transfer, and monitoring tasks (OLIVEIRA *et al.*, 2010). Examples of SWfMS are Kepler (ALTINTAS *et al.* 2004), VisTrails (CALLAHAN *et al.*, 2006), Taverna (HULL *et al.*, 2006), and Chiron (OGASAWARA *et al.*, 2013). There are other solutions such as SciCumulus, which is a middleware to orchestrate scientific workflows through SWfMS in distributed and parallel environments, such as workflow executions in cloud environments (OLIVEIRA *et al.*, 2010).

Computational support for scientific workflows through SWfMS is of particularly importance for processes that are repeatedly executed and for experiments that explore many variations on workflow instances, since these scenarios are too complex for humans to handle.

In general, SWfMS provides a procedural model to specify the workflow design. Despite this operational support, scientific users typically desire to have flexibility, to do whatever they want experiment without a tight control. Flexibility is a limitation of SWfMS, sometimes; it is difficult to provide both flexibility and support, because of conflicting requirements. So, Information System should provide balance between flexibility and support. In the Figure 2, shows in right side the part of classical workflow management systems like a SWfMS (PESIC *et al.*, 2010).

On other hand, the left-side emphasis on flexibility and user empowerment. It is difficult to visualize all possible paths and the process is driven by user decisions rather than system decisions. Groupware systems focus on supporting human collaboration, and co-decision making. They need flexibility for unpredictable result, so the expert can react to exceptional situations and execute the workflow in the different manner (PESIC *et al.*, 2010).

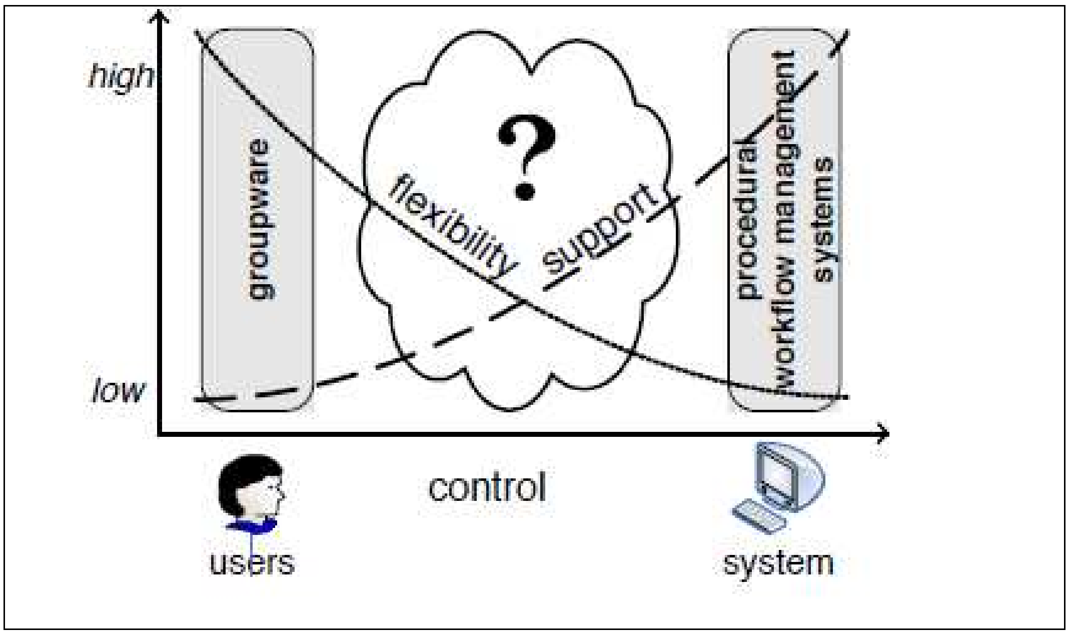


Figure 2 - Flexibility versus Support (PESIC *et al.*, 2010).

2.1 Data Provenance

Data provenance manages a collection of metadata on data origin and history. In scientific workflows, this metadata is essential for providing an operational support for scientific experiments. It comprises information about entities (input data, files, etc), activities, agents, and relationship among them. Additionally, the timestamp, parameter values, input artifacts and delivered artifacts for each activity is kept track of as provenance data (GIL *et al*., 2007)(ZENG *et al.*, 2011).

Provenance is perceived as a crucial component of scientific workflow systems, which helps scientists to ensure the reproducibility of their scientific analysis and processes, publication and contribution between coworkers (GIL *et al*., 2007), i.e., the lineage and history of results (CUEVAS-VICENTTÍN *et al.*, 2012). Provenance has been studied in various areas, such as scientific processing and databases (CHENEY *et al.*, 2009), as well as in SWfMS (LIM et *al.*, 2010). It can be queried, analyzed, visualized, mined for understanding of experiment result or workflow debug. (CUEVAS-VICENTTÍN *et al.*, 2012) (ZENG *et al.*, 2011).

Different data provenance systems have their own representation model. In order to provide interoperability among different systems, a family of specifications has been defined by W3C as a standard for provenance representation, named PROV (MOREAU; MISSIER 2013). Within PROV, the PROV-DM (PROV Data Model) is the conceptual data model specification. PROV-DM represents the relations between the entities, agent, activities, and their collection, as well as the time at which they were created (MOREAU; MISSIER 2013). Figure 3 illustrates PROV-DM as a UML diagram; the classes and the relationships among them describe the use and production of entities by activities, which may be influenced by agents and the relationships among them.

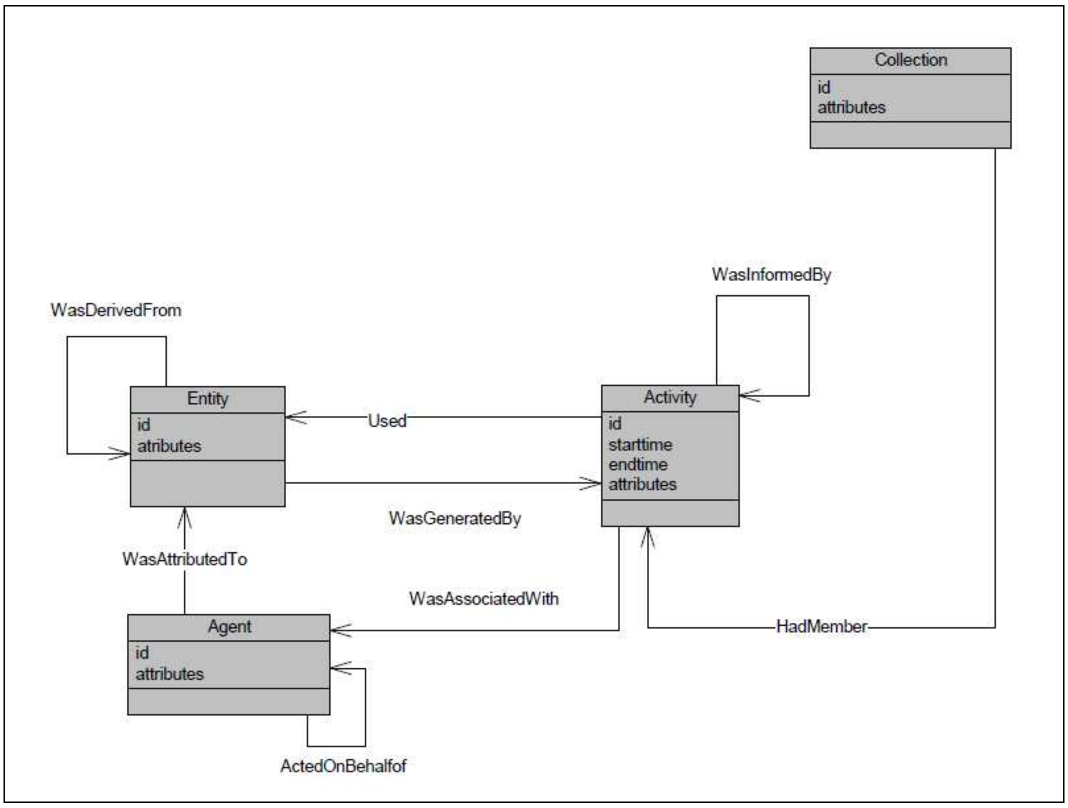


Figure 3 - PROV-DM: General overview (MOREAU; MISSIER 2013).

2.2 Scicumulus

SciCumulus is a middleware used to distribute, control and monitor parallel execution of scientific workflows on top of a SWfMS (such as VisTrail) in a cloud environment. SciCumulus orchestrates the workflow parallel execution over a distributed set of virtualized machines. SciCumulus has a distributed architecture composed by four-layer elements: desktop layer that dispatcher workflow; distribution components (execution broker, parameter sweeper, encapsulator, scheduler) that manages activities; execution modules (instance controller, configurator, executor) that perform workflow programs; and data layer (data acquisition agent, provenance database, shared filesystem) that manages data (Figure 4) (OLIVEIRA, 2012).

Scicumulus hides the complexity of the workflow parallelism in cloud environments from scientists and collects distributed provenance data following the Many Tasks Computing (MTC) paradigm, which is based on several computing resources used over short time periods in order to accomplish several computational activities. SciCumulus provides two different kinds of parallelism functionalities: parameter sweep and data parallelism. Following the definition of a workflow W ({a_i},

…{a_j} previously described; parameter sweep is a parallelism functionality in which each activity a_i is performed using configured parameters pt_{ij}, and for each parameter pt_{ij} an instance of a_i is performed. Data parallelism, on the other hand, may be depicted as simultaneous execution of an activity a_i that consumes a subset d_{ij} of the input data for a_i. For each input subset d_{ij}, one instance of ai is executed using d_{ij} as its input. In other words, the same activity is executed with different input data.

Scicumulus captures provenance data dynamically, during the execution of the scientific workflow. Therefore, the scientist is able to gather and query provenance data during workflow execution and analyze the experiment results. Provenance data is inserted into a W3C PROV-compliant database and persisted in PostgreSQL RDBMS (Relational Database Management System) (OLIVEIRA *et al.*, 2010) (COSTA *et al.*, 2013).

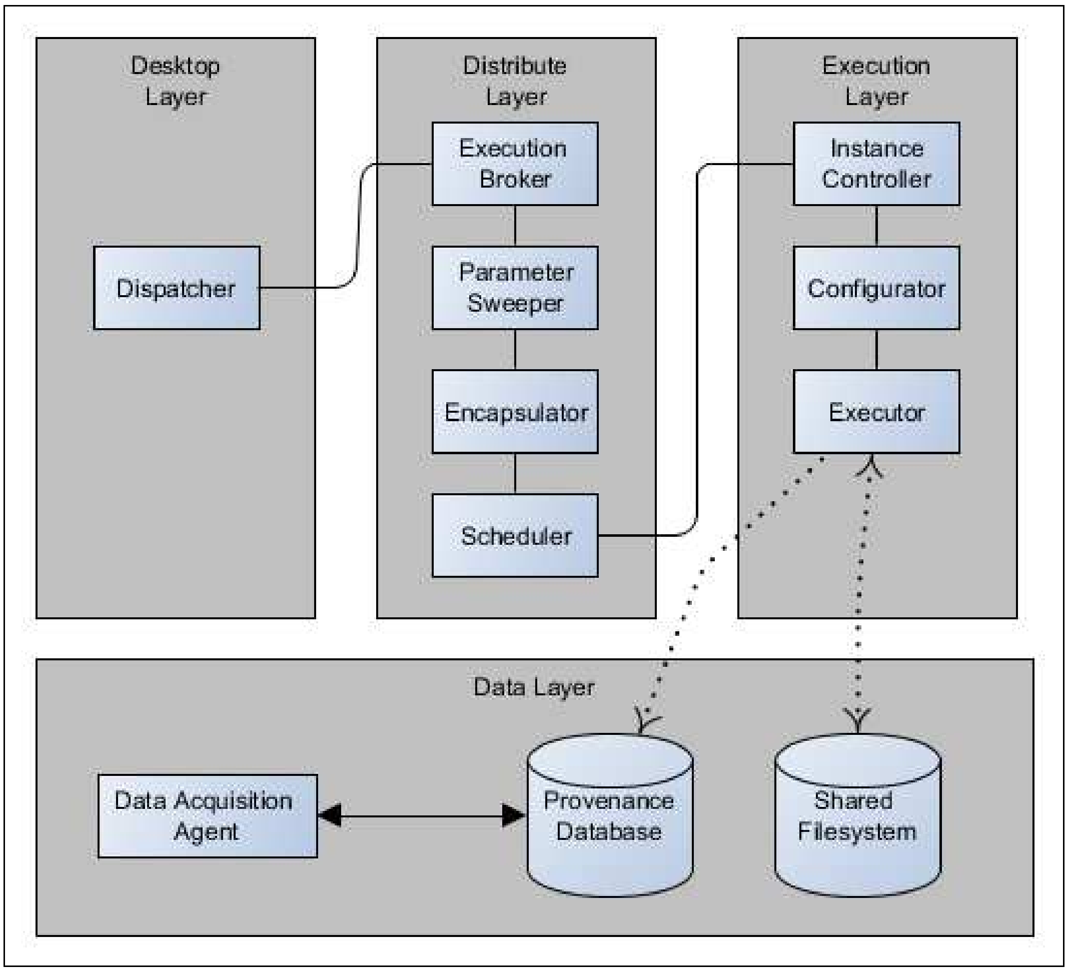


Figure 4 - Scicumulus Conceptual Architectural (Adapted from OLIVEIRA, 2012)

Chapter 3 – Declarative Process Mining

Process mining provides a link between data mining and BPM (Business Process Management). It is an approach for discovering, monitoring and improving real processes through extracting knowledge from event logs acquired by information systems. Process mining may be applied to automated process discovery; conformance checking (comparing model and logs); simulation models construction; and history-based recommendations (VAN DER AALST *et al*., 2012)

There is a plethora of process modeling languages, which may be grouped into imperative [such as BPMN (WHITE, 2004), Petri nets (MURATA, 1989), UML ADs (STOERRLE)] or declarative [Declare (MAGGI *et al.*, 2011), DCR Graphs (HILDEBRANDT, 2011)]. While the former models all possible steps of a process, the latter focus on the logic that governs interactions between the actions of a process, describing what can be done by restricting only the undesired behavior (ZUGAL *et al*., 2013).

Imperative modeling specifies the procedure of how process has to be executed, thus requiring all possible process paths to be explicitly specified in the model before the workflow execution. Every new step must be added to the model during experiment specification in experiment composition phase. In contrast, declarative process modeling does not specify the control-flow of activities *a priori*. Instead of determining all possible process paths, only its essential characteristics are described through rules. Adding new constraints to the model limits the number of workflow execution alternatives. This way, every execution control flow is possible, as long as it does not violate any of the specified constraints (VAN DER AALST *et al.*, 2009).

Declarative models describe a process in an “open world”, while procedural models is a “closed world”. Figure 5 illustrates the difference among the universe of possible, forbidden, optional and allowed paths for a process execution following either a procedural model (a more traditional approach) or a declarative model (constraint-based approach) which enhances flexibility.

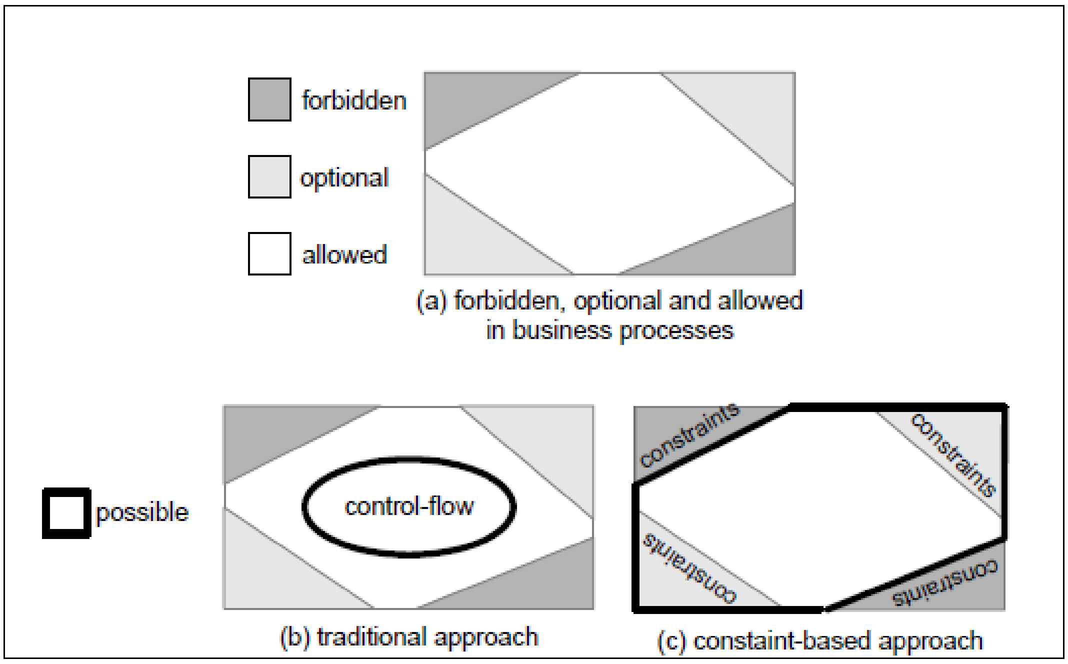


Figure 5 - Declarative vs. Procedural Models (W.M.P. VAN DER AALST *et al.*, 2009)

Flexible models allow increasing the user decision making, moving choices from workflow design time to run time. To support users in making decisions, the information systems may provide recommendations through process mining techniques. These recommendations may depend on explicit domain knowledge, but it is possible to learn from a process with past workflow execution and then give to user for insight in planning future experiment scenarios. The expert may decide to discard the recommendation, but at least some decision support is given. The analysis of workflow instances becomes more relevant when experts are not forced to work in a particular way (VAN DER AALST *et al.*, 2009).

3.1 Declare

Declare is a process modeling language based on the declarative paradigm. Declare maps are interesting in the context of process mining. One can discover Declare maps from event logs (extracted from audit trails, transaction logs, and databases) without preexisting models and knowledge. Declare provides flexibility mechanisms, such as: defer (decide to decide later), change (decide to change model), and deviate (decide to ignore model) (VAN DER AALST *et al.*, 2009).

Declare model presents recommendations to users as independent information, the users does not compel to follow recommendation. It offers desirability rules to execute tasks, re-do tasks that were executed before or even skip tasks that should be executed users (PESIC *et al.,* 2007).

Declare is an open source (WESTERGAARD; MAGGI, 2011) LTL (Linear Temporal Logic) rule collection, which shows the control-flow dependency between two activities on finite traces through graphical representation (MAGGI *et al.*, 2103). It allows the discovery of non-sequential or coexisting events in the same process instance. The language is intended to be understandable for end-users. The LTL operator O mean "has to hold in the next position of a path", while operator □ has a semantic "has to hold always in the subsequent positions of a path" and operator ♦ means "has to hold eventually (somewhere) in the subsequent positions of a path" (Table 1) (MAGGI *et al.*, 2011).

Table 1 - LTL Operator semantics (MAGGI et al. 2011)

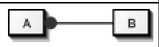
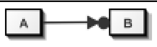
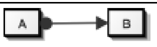
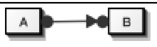
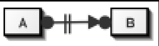
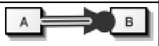
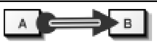
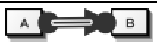
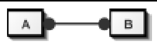
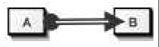
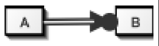
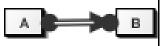
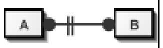
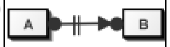
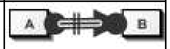
Operator	Semantics
O	has to hold in the next position of a path.
□	has to hold always in the subsequent positions of a path
♦	has to hold eventually (somewhere) in the subsequent positions of a path

Declare describes a set of constraints which must be satisfied throughout the process execution. The constraints are classified in templates in the groups: existence, relation, negative relation, and choice (MAGGI *et al.*, 2011).

This task of automatically learning a declare model from data is known as Declare mining. An implementation for declare mining is available as plug-in in Prom tool. It uses constraint templates with a graphical notation and implements semantics through operations such as init (A), precedence(A,B), response(A,B), succession(A,B), not succession(A,B), chain succession (A,B), co-existence(A,B), among others (MAGGI *et al.*, 2011).

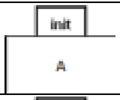
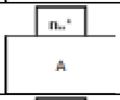
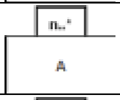
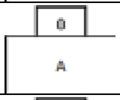
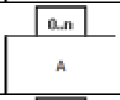
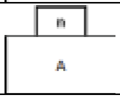
A relation template states a dependency among two activities. For example, the "co-existence(A,B)" template states that if one of the events A or B occurs, the other one should also occur. The templates: "chain response", "chain precedence" and "chain succession" determines that the occurrences of the two activities (A and B) are next to each other (MAGGI *et al.*, 2011). Table 2 depicts constraint templates in Declare, along with their meaning, LTL semantics, and graphical representation

Table 2 -Declare Constraint Templates (Adapted from MAGGI *et al.*, 2011; BOSE *et al.*, 2013)

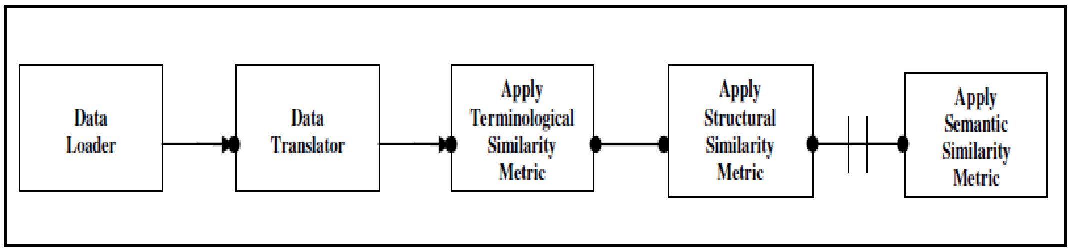
Constraint	Meaning	LTL semantic	Graphical notation
responded existence	if A occurs then B occurs before or after A	♦A → ♦B	
precedence(A,B)	if B occurs then A occurs before B	(¬B UA) ∨□ (¬B)	
response(A,B)	if A occurs then eventually B occurs after A	□ (A □ ♦B)	
succession(A,B)	for A and B both precedence and response hold	response(A,B) ∧ precedence(A,B)	
not succession(A,B)	if A occurs then B cannot eventually occur after A	□ (A □ ¬ (♦B))	
chain precedence(A,B)	if B occurs then A occurs in the next position before B	□ (B → O A)	
chain response(A,B)	if A occurs then B occurs in the next position after A	□ (A → O B)	
chain succession(A,B)	for A and B both chain precedence and chain response hold	□ (A → O B) ∧ □ (B → O A)	
co-existence(A,B)	if A occurs then B occurs before or after A and vice versa	♦A ↔ ♦B	
alternate response	if A occurs then eventually B occurs after A without other occurrences of A in between	□ (A → O (¬A U B))	
alternate precedence	if B occurs then A occurs before B without other occurrences of B in between	(¬B U A) ∨□ (¬B) ∧ □ (B → O (¬B U A) ∨□ (¬B)	
alternate succession	for A and B both alternate precedence and alternate response hold	alternate response(A,B) ∧ alternate precedence(A,B	
not co-existence	A and B cannot occur together	¬ (♦A∧♦B)	
not succession	if A occurs then B cannot eventually occur after A	□ (A → □ ¬ (♦B))	
not chain succession	if A occurs then B cannot occur in the next position after A	□ (A → O (¬B))	

The existence templates describe a unary relationship and define the cardinality or the position of an event in a process instance, such the template init(A) of a activity specify the process instances start with activity A. Table 3 shows the existence templates with its meaning, graphical notation and LTL semantics. The existence(n,A) specify that A should occur at least n times in a process instance. While, templates of the type absence(n+1,A) specify that A should occur at most n times. Templates exactly(n,A) indicate that A should occur exactly n times.

Table 3 -Existence Templates (Adapted from MAGGI et al. 2011)

Constraint	Meaning	LTL semantic	Graphical notation
init(A)	start with A	A	
existence(1,A)	A should occur at least one time	♦A	
existence(n,A)	A should occur at least n times	♦(A ∧ O (existence (n-1, A)))	
absence (A)	A should occur at most one time	¬ existence (1, A)	
absence(n,A)	A should occur at most n times	¬ existence (n+1, A)	
exactly(n,A)	A should occur exactly n times	existence (n,A) ∧absence (n+1, A)	

The figure 6 shows a learned declare model extracted from an ontology process (SILVA *et al.*, 2014). The learned rules state that "data translator" activity only occurs after "data loader" activity, "apply terminological similarity metric" and "apply structural similarity metric" co-exist, meaning that if the scientist plan to apply a terminological metric in a future scenarios, the structural metric should also be applied and "apply semantic similarity metric" activity cannot co-exist with "apply structural similarity metric", meaning that the scientist should plan future scenarios choosing between applying structural or semantic metrics, but not both.


Data Loader
Data Translator
Apply Terminological Similarity Metric
Apply Structural Similarity Metric
Apply Semantic Similarity Metric
Figure 6 - Declarative model of scientific experiment of ontology matching (SILVA et al. 2014)

A discovered Declare map has many constraints that are redundant, it may be pruned by selecting only those constraints that are the most interesting for the user, so, we can simplify the model without lose meaning. Figure 7 presents a declare constraint hierarchy, the solid arcs indicate which constraints dominate other constraints. A constraint can be discarded if there is a directed path to it from another constraint involving the same event. For example, a constraint succession(A, B) is redundant if a stronger constraint chain(A,B) holds. The dashed arcs indicate constraints that are transitive (MAGGI *et al.*, 2013b).

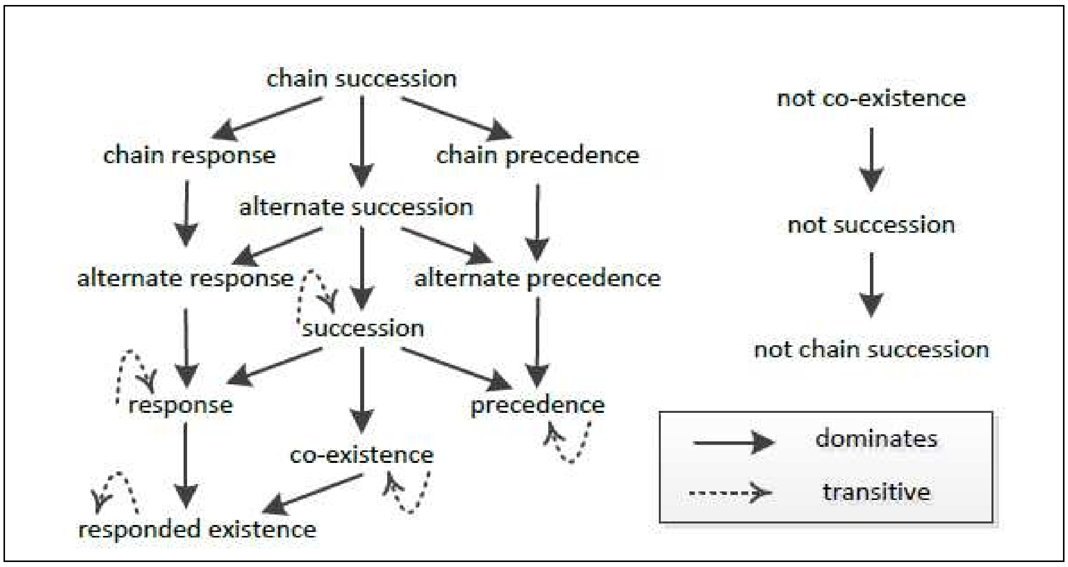


Figure 7 - Declare constraint hierarchy (MAGGI *et al.*, 2013b)

3.2 Validation declarative models

It is not trivial to evaluate a declarative model, because is no defined to point precisely deviations and quantify discrepancies in a model with absence notion of state. Mainly, in order to analyzing the compliance of a constraint-base model some constraints can be vacuously satisfied. BURATTIN *et al.* (2012) introduce the notion of healthiness of a trace, based on the concept of activation of a declare constraint.

A constraint activation occurs when an occurrence of activity forces an behavior in process in relation of other activity. For example, if a constraint *not co-existence(A, B)* is activated, means that the occurrence of activity A forces not occurrence of the activity B. A constraint activation may be fulfilled or violated, following the constraint example above, if activities A and B did not occur in same process instance, then the activation was fulfilled. However, if activities A and B happened in same process instance, then the constraint was violated. The metric that measures the quantity of fulfilled constraint activation is called *fulfillment*, on other hand; *violation* is a metric that measures the quantity of violated constraint activation. The ratio of fulfillment of a constraint over total number of activation defines the metrics *fulfillment ratio* and *violation ratio* (BURATTIN, 2012; MAGGI *et al.*, 2013). i.e.:

fulfillment ratio = Σ fulfillment / activations

violation ratio = Σ violation / activations

Chapter 4 – Proposed Approach

In this chapter, we describe our proposed approach for combining the data-attribute perspective with the control-flow perspective, resulting in a data-aware declarative model discovered from provenance data of scientific experiments. Thus, scientists can have the balance among flexibility and support in execution of scientific experiments. Due to the flexibility, the learned data-aware declarative model provides insight to scientists when planning new workflow scenarios, since they will be faced with combinations of parameters and activities that frequently resulted in successful workflow instances, based on user-requirements and executed in a traditional SWfMS. Moreover, the scientists will have in their hands the operational support provided by a SWfMS (with predefined flow of activities) and a flexible declarative model to aid them in making decisions to explore new workflow scenarios that tend to be more successful, to avoid those scenarios that tend to be unsuccessful, and even to better understand the reasons for both.

Figure 8 shows a high level view of the main components of our proposed approach. Initially, the scientists specify their workflow templates and execute several scenarios – varying algorithms, parameters and resources – using the conventional SWfMS platform that they are used to. Those executions generate provenance data. After several executions of workflow, the provenance data collected from historical executions is filtered, based on user requirements, generating a new data subset. This subset of provenance data is used to automatically build a data-aware declarative model. The scientist then analyzes this model and identifies which were the most appropriate alternatives for activities and parameter values, fine-tunes the workflow template and plans new scenarios based on this information, and finally (re)executes these new scenarios. This process is interactive and incremental, until the scientists validates or refutes their research hypothesis.

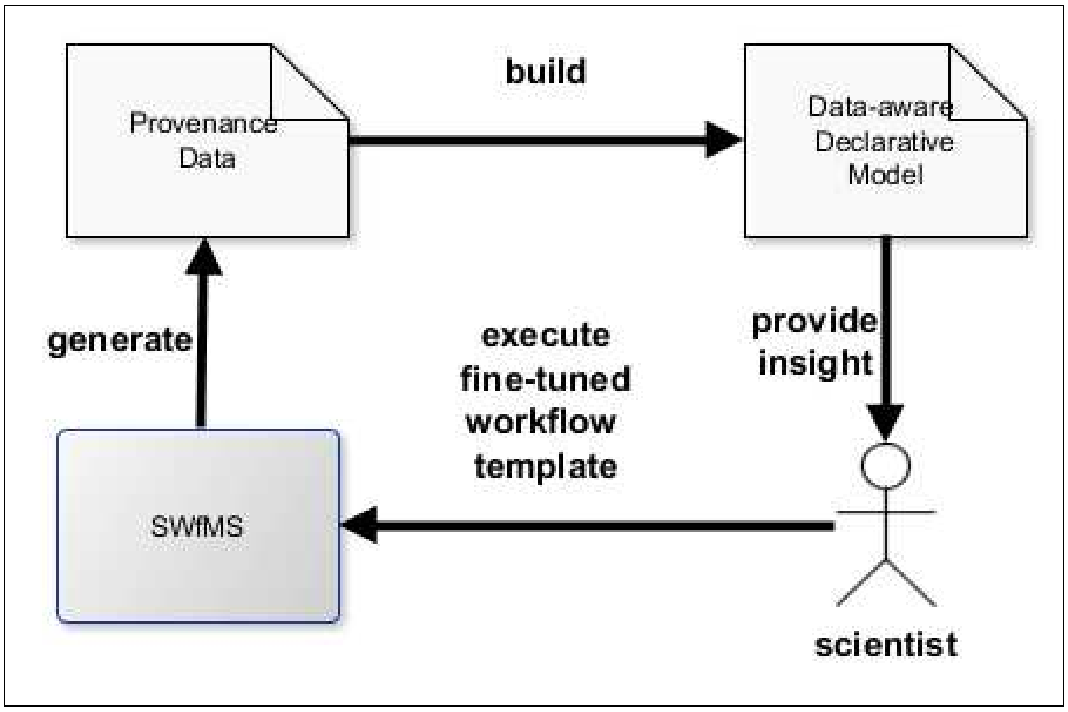

build
Provenance Data
Data-aware Declarative Model
generate
execute fine-tuned workflow template
provide insight
SWfMS
scientist

Figure 8 - Overview of the main components of the proposed approach

The second view of the approach (Figure 9) illustrates the three main steps proposed by our proposed approach: “User Requirement Definition”, “Workflow Execution”, and “Data-aware Declare Model Learning”. The first step (“User Requirement Definition”) should be executed by the scientist, and encompasses the definition of experiment metric(s). Those metrics represents and operationalizes the user requirements on the quality of their scientific workflow results, thus reflecting how the results of a particular workflow instance should be evaluated. For example, a metric “precision” more than 0.8 can be define the workflow instance quality. The metrics should be defined by the scientist, who also is responsible for implementing an activity to automate the calculation of each metric, and attach this activity to the scientific workflow template. These implemented metrics allow an automatic classification of each workflow instance as either being successful or not, based on scientist requirements. The metrics defined by the scientist are stored in the provenance database. For example, in a data mining experiment the scientist may define precision, recall and f-measure as metrics, implement a service to calculate these values, and include a workflow activity as the last step of the data mining workflow, invoking this service.

In the Workflow Execution step, the SWfMS should be already configured to access a PROV-compliant database (extended with the metrics tables we propose). The scientist is responsible for defining workflow scenarios they wants to execute, specifying each scenario in the SWfMS and executing each scenario. During each execution, an engine implemented on SWfMS collected and stored provenance data in a database. For example, in the data mining experiment may be specified and executed on top of VisTrails. VisTrails collects the provenance data of each workflow instance, as well as its user-defined metrics.

The last step, "Data-aware Declare Model Learning", is the main step of our approach, and is detailed in Figure 10. The learning (discovery) process encompasses four activities: "Provenance Data Filtering According to Chosen Metrics", "Data-aware Rules Discovery", "Provenance Data Filtering According to Data-aware Rules" and "Data-aware Declarative Model Discovery", which will be detailed in the following sections.

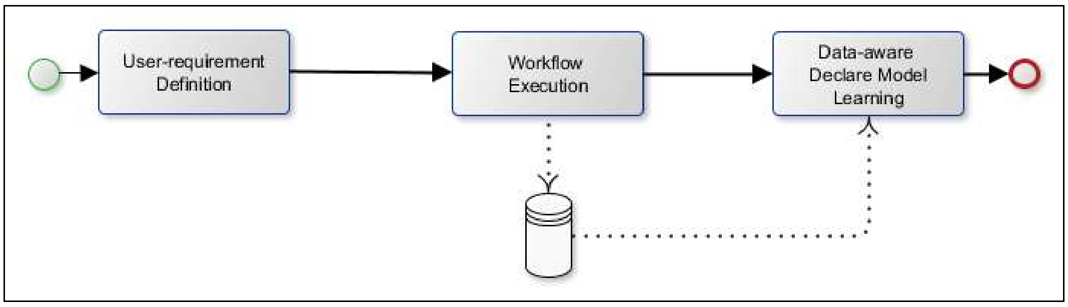

User-requirement Definition
Workflow Execution
Data-aware Declare Model Learning

Figure 9 - Proposed Approach Steps.

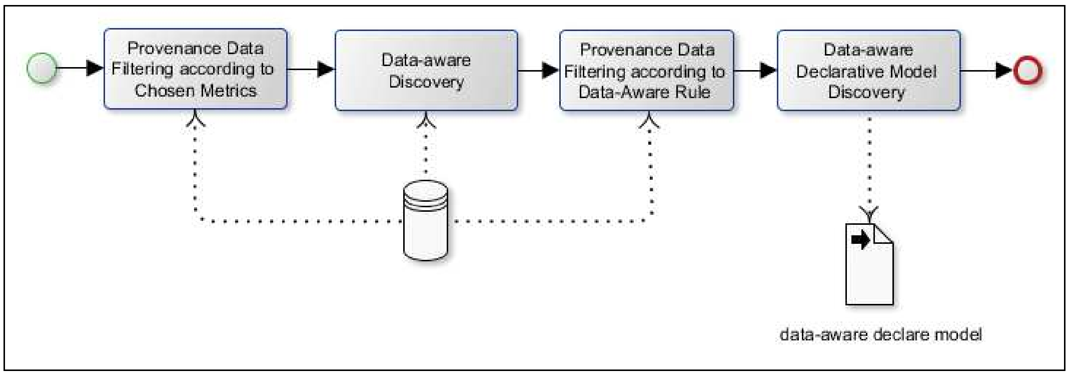

Provenance Data Filtering according to Chosen Metrics
Data-aware Discovery
Provenance Data Filtering according to Data-Aware Rule
Data-aware Declarative Model Discovery
data-aware declare model

Figure 10 - Data-Aware Declarative Model Learning activities

4.1 Provenance Data Filtering According to User-Requirements

In a detailed view for learning a data-aware declarative model (Fig. 8), the first step is "Provenance Data Filtering According to User-requirements". This activity filters the provenance data based on workflow instances whose log contains information about the metrics chosen by the scientist. We extended PROV-DM metamodel to persist pre-defined metrics and capture their values. The PROV-DM metamodel provides flexibility for not previously defined entities, so it suited for the specification of several specific user-defined requirements. Figure 11 shows the extended data model of PROV-DM, including “User-defined Metric” Entity (blue color), that qualifies objects like “Activity” (or even “Collection” of activities) and/or “Entity”.

Once the metric is collected for each workflow instance, in this activity the workflow instances that satisfy user-requirements are collected, thus filtering provenance data to only consider the instances one is interested in analyzing. For example, a scientist may be interested in analyzing workflow instances that overcome 0.8 in precision (due to insights that they may constitute scenarios that will lead to good results) or, yet, only instances with precision less than 0.8 (in order to learn which scenarios should be avoided). Thus, the provenance data to be mined into the declarative model is based on user-requirements which are inserted in a metamodel that extends the PROV-DM.

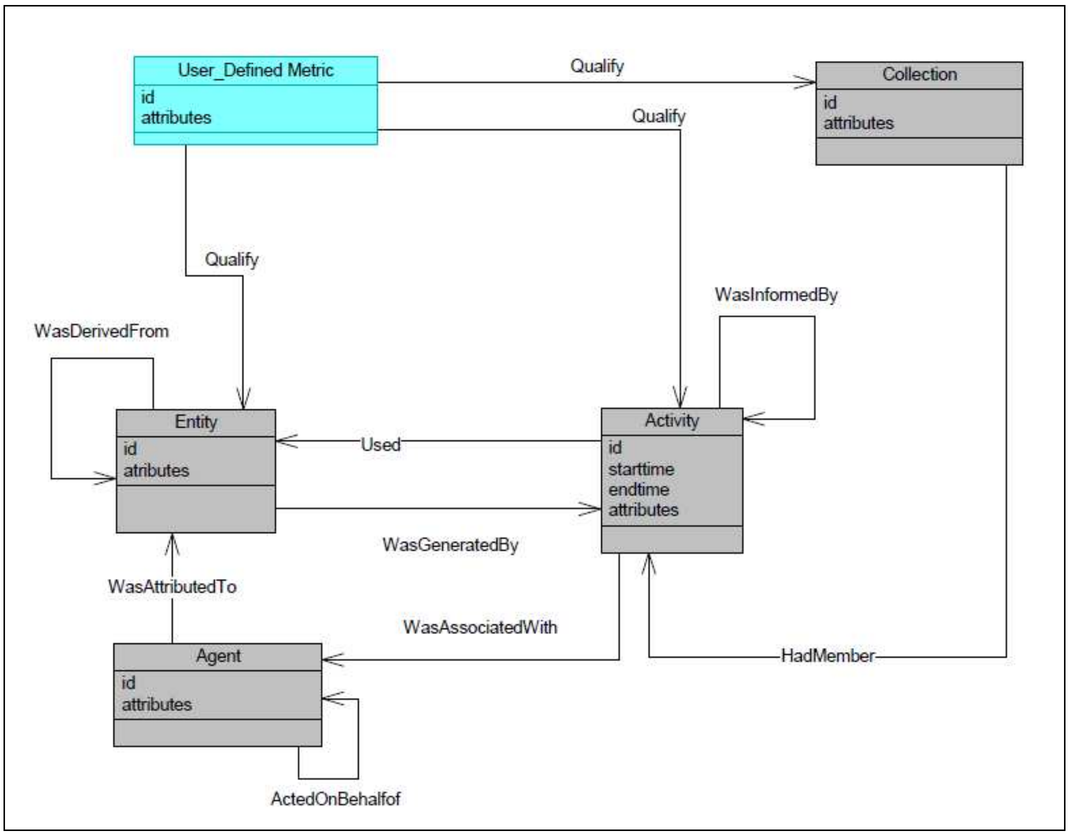


Figure 11 - Extended PROV-DM

4.2 Data-Aware Discovery

The second step is “Data-Aware Discovery”, which discovers a classification model from the provenance data subset produced by the previous activity. The idea is to learn data-aware constraints that are recurrently consistent with the instances logged in the provenance data. Decision tree was chosen as the classification model, since it fits nicely to represent constraint rules. Structurally, each branch in the decision tree corresponds to a conjunction of data constraints (in the form <data attribute><operator><value>, for example “pα> 2 AND pβ <9”) that filters provenance data. The classification algorithm generates a decision tree whose class attribute indicates whether the workflow instance is successful or not. Figure 12 depicts a generic decision tree.

4.3 Provenance Data Filtering According to Data-aware Rules

The scientist then chooses a branch according to its interest reflected in the class attribute. Then, in the third step, “Provenance Data Filtering According to Data-aware Rules”, the constraints are represented by this branch are transformed in a set of filters that are applied in the provenance database through the use of a SQL query, deriving a new provenance filtered data. For example, the query “SELECT case, activity_name, start_time FROM activity” is appended with filter “WHERE parameter1>=value AND parameter2>=value AND parameterN= “value”.

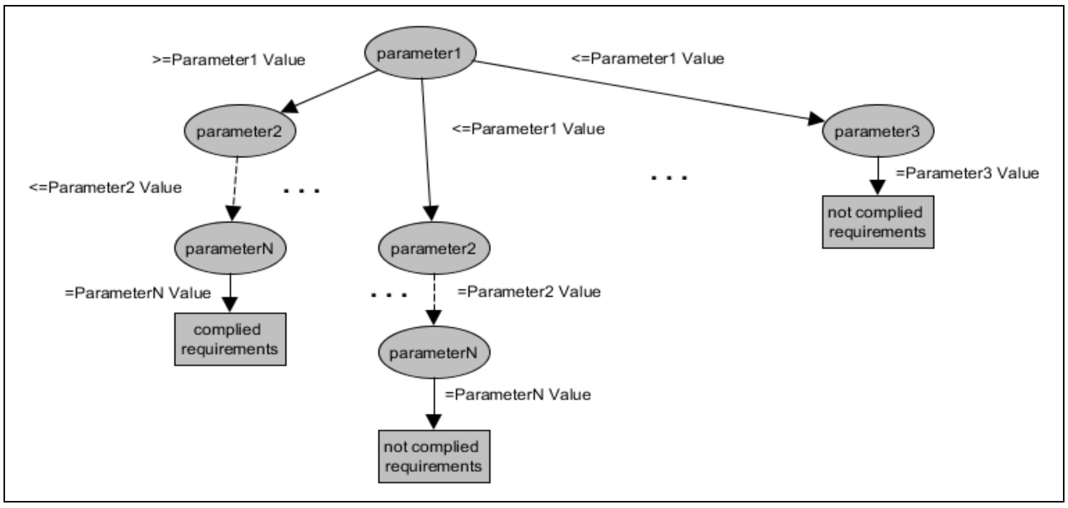


Figure 12 - Generic Decision Tree learned from Provenance Filtered Data

4.3 Data-Aware Declarative Model Discovery

The last activity is “Data-Aware Declarative Model Discovery”, which is responsible for running an algorithm for declarative model discovering considering the provenance filtered data found in third activity. Since the declarative model found was learned based on data filtered from data-aware constraints, we call it a data-aware declarative model.

In order to represent data-aware declarative models, we extended declare constraint templates, mentioned in section 3.1, with data-aware constraints. These data-aware constraints correspond to the decision tree branch chosen by the user. Table 4 presents the formal semantics of some individual data-aware declare constraints templates proposed. In the proposed templates, each constraint <data-attribute><operator><value> is represented by a term named $cond_i$. The conjunction of several data constraints $cond_i$, … $cond_j$, i < j, is represented as $\wedge cond_{i\ j}$. Therefore,

$$\wedge cond_{i\ j} = cond_i \ \wedge\ cond_{\dots}\ \wedge\ cond_j$$

For example, $\wedge cond_{i\ j}$, init(A), response (A,B); it means that if parameter values of i to j were satisfied the workflow should start with activity A and activity B occurs occasionally after A.

The graphical representation for the proposed data-aware declare constraint is given as a precedent symbol connected to the declare diagram by an edge labeled “$\wedge$condi j”. An example using this notation is illustrated in Figure 13.

Table 4 - Data-aware Declare constraints

Data-aware constraint	Meaning
$\wedge cond_{i\ j}$, init(A)	if ($cond_i$ $\wedge$…$\wedge cond_j$) then (start with A)
$\wedge cond_{i\ j}$, precedence(A,B)	if ($cond_i$ $\wedge$…$\wedge cond_j$) then (if B occurs then A occurs before B)
$\wedge cond_{i\ j}$, response(A,B)	if ($cond_i$ $\wedge$…$\wedge cond_j$) then

	(if A occurs then eventually B occurs after A)
$\wedge cond_{i\ j}$, succession(A,B)	if ($cond_i \wedge ... \wedge cond_j$) then (for A and B both precedence and response hold)
$\wedge cond_{i\ j}$, chain precedence(A,B)	if ($cond_i \wedge ... \wedge cond_j$) then (if B occurs then A occurs in the next position before B)
$\wedge cond_{i\ j}$, chain response (A,B)	if ($cond_i \wedge ... \wedge cond_j$) then (if A occurs then B occurs in the next position after A)
$\wedge cond_{i\ j}$, chain succession(A,B)	if ($cond_i \wedge ... \wedge cond_j$) then (for A and B both chain precedence and chain response hold)
$\wedge cond_{i\ j}$, co-existence(A,B)	if ($cond_i \wedge ... \wedge cond_j$) then (if A occurs then B occurs before or after A and vice versa)

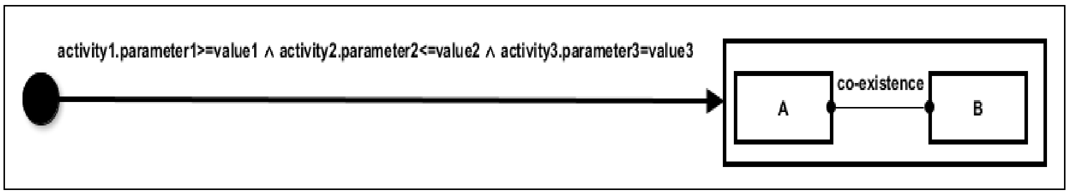

activity1.parameter1>=value1 ∧ activity2.parameter2<=value2 ∧ activity3.parameter3=value3
co-existence
A
B

Figure 13 - Data-aware Declare Diagram.

4.4 Solution Architecture

Our proposal is supported by the technological architecture elements and artifacts depicted in Figure 14. First, the Provenance Database is created by instantiating the extended PROV-DM schema in a relational database management system. In workflow design time, the scientist models the workflow template in the SWfMS interface, and includes additional workflow tasks to gather user-defined metrics as provenance data. Then, in workflow execution time, an initial set of workflow instances is executed, populating the provenance database. A dataset is acquired by executing SQL query with filters, next, a decision tree learning algorithm is executed through a Data Mining Tool in order to discovery a set of data constraints. Finally, the Declarative Miner Tool for discovering a data-aware declare model is executed considering the filtered data found by applying the constraints discovered. The steps to send data for Data Mining Tool and Declare Miner Tool is made manually by the user.

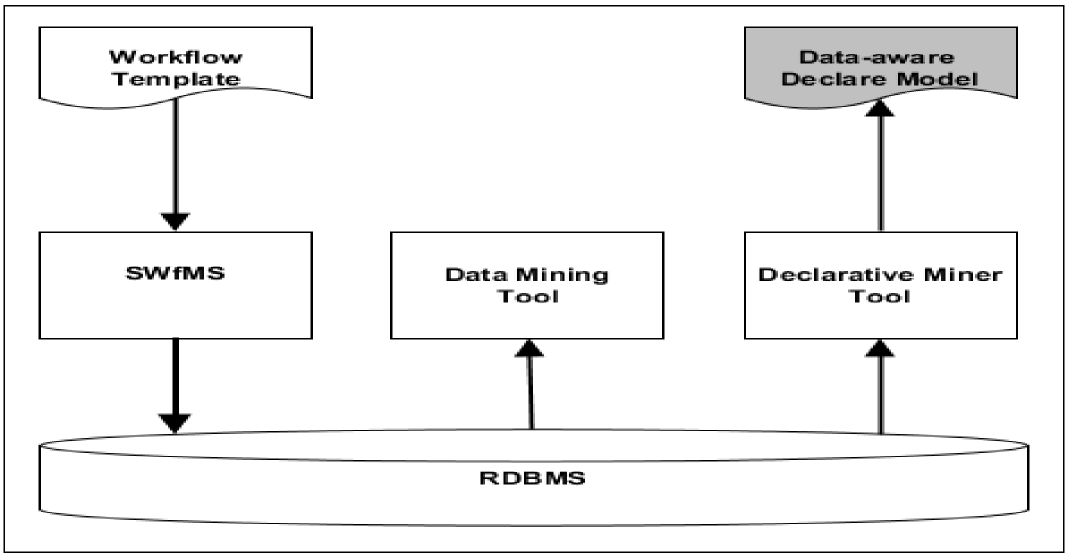


Figure 14 - Solution Architecture

Chapter 5 – Use Scenarios

In order to measure the accuracy of the data-aware declare model, there are metrics proposed in BURATTIN *et al.* (2012). These conformance metrics evaluates the degree of healthiness of a process trace and of a log, using indicators such as activation, fulfillment, violation and conflict for a declare constraint.

The "Declare Analyzer" plug-in from the ProM framework was used to quantify the degree of adherence of each trace in terms of number of fulfillments and violations ratio (BURATTIN *et al.* 2012). These metrics are percentages of violations and fulfillments of the constraints over the total activations (MAGGI *et al.*, 2013b).

This chapter describes in detail the evaluation of our proposed data-aware declarative model learned on top of provenance data. The evaluation consists of two experiments aiming to observe how precise the combined models are. The first is an exploratory user scenario data mining experiment for evaluating the potential of approach. The second was applied on a real scenario for evapotranspiration estimation.

5.1 Wish Detection Experiment

In this section, we present a use scenario of data mining experiment, specifically for classifying training algorithms on the task of detecting wishes in tweets (GONÇALVES *et al.*, 2015). The training configuration used the Wish Corpus as training data and NLTk 3.0 on Python 2.7.1 as the framework for the implementation of the Naive Bayes algorithm. The abstract workflow is depicted in Figure 15 in a BPMN diagram.

The workflow has eight activities; some of them with varying parameters, such as: "corpus" in activity 1, word length in activity 4 and the partition size in activity 6. Table 5 depicts the activity name, its description, and parameters.

The modeling and execution of scientific workflow in SWfMS was part this work. The workflow template was modeled and executed through the SciCumulus SWfMS. Provenance data was collected during workflow executions and stored in a PostgreSQL relational database. The experiment was executed in a cloud environment, the Amazon EC2 platform, using Linux instances (large server and micro instances).

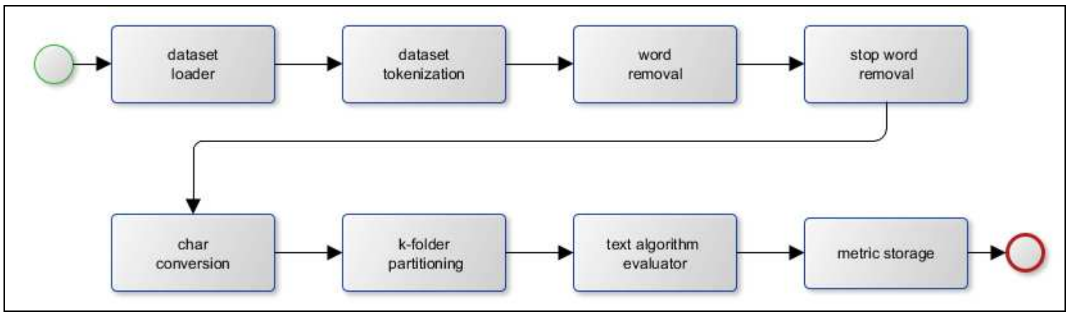


Figure 15 - Wish Detection Workflow

Table 5 – Wish Detection Workflow Activities

Activity no.	Activity Name	Activity Description	Parameter	Parameter Description	Domain Data
1	dataset loader	load corpus	corpus	wish corpus	politics, products, politics_products
2	dataset tokenization	divide a text into a list of sentences, by using an unsupervised algorithm.	NA		
3	char conversion	convert all word in lower case	NA		
4	word removal	remove word in according to length configured in parameter lword	lword	length of word to be removal	0-n
5	stop word removal	remove stop words, such as: the, is, at, which, and on	NA		
6	k-folder partitioning	partition the database in k-folders	kfold: 1-n	number of fold partitioning	1-m
7	text mining algorithm	perform an algorithm for text mining	NA		
8	metric storage	store metrics of workflow instance	NA		

The user-requirements for filtering provenance data were set by the scientist as follows: Precision >0.6 was the chosen metric for analyzing workflow instances. Therefore, a clause “WHERE metric_name = ’pos-precision’ was appended to SQL query to generate a filter provenance data. In order to discover data-aware constraints, the Random Tree algorithm from the Weka toolbox (HALL et al., 2009) was chosen as classification learning algorithm. Figure 16 depicts the discovered decision tree.

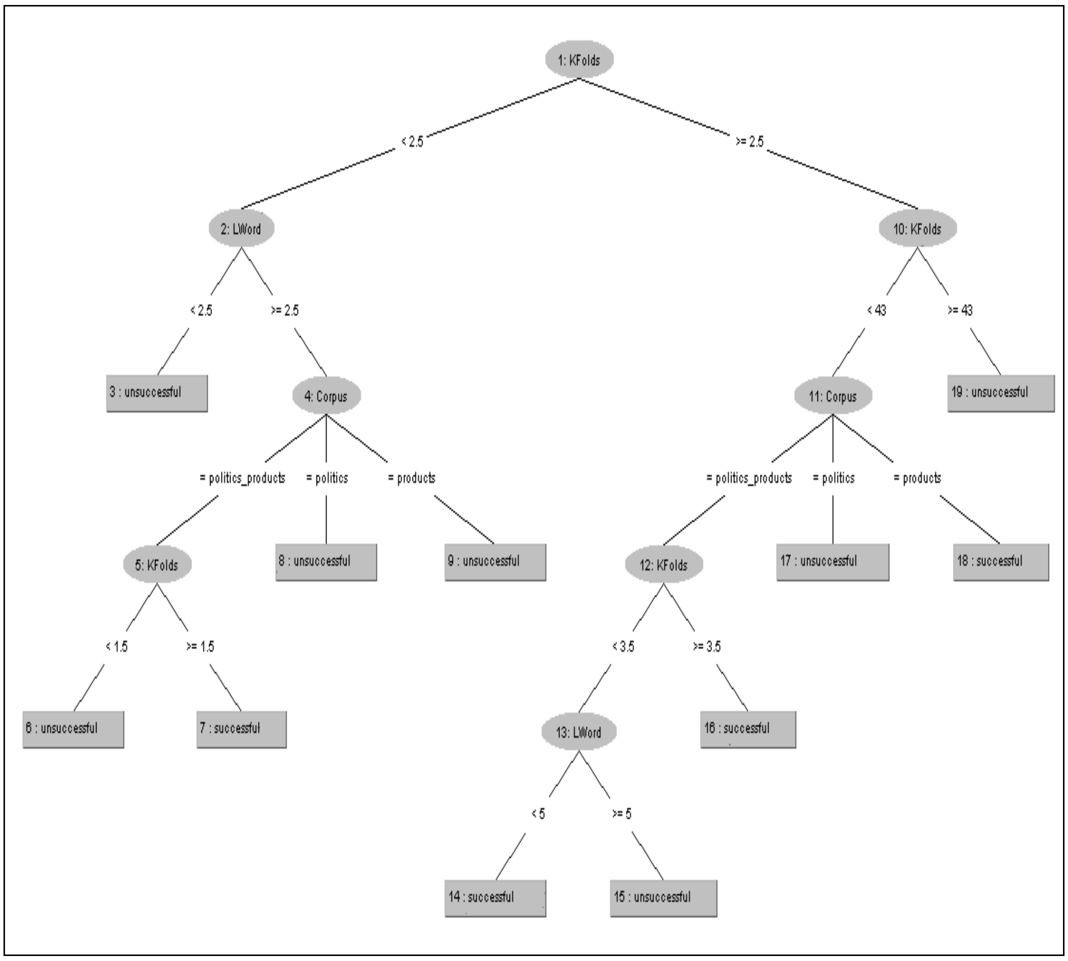


Figure 16 Discovered classification tree

The leaves with the class attribute values detailed the parameter values achieved. In this example, the scientist was interested on evaluating instances with a precision greater than 0.6, thus one or more three branches can be considered:

- $\wedge \text{cond}_{i\ j}$ = kfold<=2.5 and lword>=2.5 and corpus= “politcs_products”.
- ∧condij= kfold>=1.5 and kfold>=2.5 and lword>=2.5 and corpus= “politcs_products”.
- ∧condi j = kfold>=2.5 and kfold< 43 and corpus= “products”.
- ∧condij = kfolds >3.5 and kfolds <43 corpus = “politics_products”
- ∧condij= kfolds>3.5 and kfolds<43 and lword <5 and corpus = “policts_products”

So, each of these conditions was appended to a SQL query in a WHERE clause for finding the cases that used the selected parameters. An example of SQL query considering filters with parameter values is presented below.

```
SELECT  w.wkfid AS "Case ID",a.tag AS "Activity",
      a.starttime AS "Start Timestamp", a.endtime AS "End Timestamp"
FROM parameter p INNER JOIN hactivity a
          ON (p.actid=a.actid)
          WHERE kfold <=2.5 AND lword >=2.5 AND corpus = ‘politics_products’;
```

After applying each constraint, the filtered provenance data was sent as input data to the Declarative Process Mining Tool. In this use scenario, we adopt DeclareMiner plug-in (MAGGI et al. 2011) in ProM tool as the declarative process mining tool.

Due to simplicity for illustration purposes, the extracted model considered only init, succession, chain succession and co-existence declare constraints. The 10 discovered data-aware declare constraints are described in Table 6.

Table 6 - Data-aware Declare Constraints

Rule1	DataSetKFolderPartitioner.kfold>=3.5 ∧ DataSetWordRemoval.lword< 5 ∧ DataSetTraining.corpus=politics_products, init (“Dataset Training Loader”)
Rule2	DataSetKFolderPartitioner.kfold>=3.5 ∧ DataSetWordRemoval.lword< 5 ∧ DataSetTraining.corpus=politics_products, chain succession (“Dataset Training Loader”, “Dataset Tokenization”)
Rule3	DataSetKFolderPartitioner.kfold>=3.5 ∧ DataSetWordRemoval.lword< 5 ∧ DataSetTraining.corpus=politics_products, chain succession (“Dataset Training Tokenization”, “Dataset Training Char Conversion”)

Rule4	DataSetKFolderPartitioner.kfold>=3.5 ∧ DataSetWordRemoval.lword< 5 ∧ DataSetTraining.corpus=politics_products, chain succession (“Dataset Training Char Conversion”,“Dataset Training Word Removal”)
Rule5	DataSetKFolderPartitioner.kfold>=3.5 ∧ DataSetWordRemoval.lword< 5 ∧ DataSetTraining.corpus=politics_products, succession (“Dataset Training Word Removal”,“Dataset Training Special Word Removal”)
Rule6	DataSetKFolderPartitioner.kfold>=3.5 ∧ DataSetWordRemoval.lword< 5 ∧ DataSetTraining.corpus=politics_products, succession (“Dataset Training Word Removal”,“Dataset Training Stop Word Removal”)
Rule7	DataSetKFolderPartitioner.kfold>=3.5 ∧ DataSetWordRemoval.lword< 5 ∧ DataSetTraining.corpus=politics_products, co-existence (“Dataset Training Special Word Removal”,“Dataset Training Stop Word Removal”)
Rule8	DataSetKFolderPartitioner.kfold>=3.5 ∧ DataSetWordRemoval.lword< 5 ∧ DataSetTraining.corpus=politics_products, succession (“Dataset Training Special Word Removal”,“Dataset Training KFolder Partitioner)
Rule9	DataSetKFolderPartitioner.kfold>=3.5 ∧ DataSetWordRemoval.lword< 5 ∧ DataSetTraining.corpus=politics_products, succession (“Dataset Training StopWord Removal”,“Dataset Training KFolder Partitioner
Rule10	DataSetKFolderPartitioner.kfold>=3.5 ∧ DataSetWordRemoval.lword< 5 ∧ DataSetTraining.corpus=politics_products, chain succession (“Dataset Training KFolder Partitioner”, “Naive-Bayes Evaluator”)

The learned data-aware declarative model presents knowledge combining constraints on events flow and activities parameters. Figure 17 depicts the extracted model (considering only init; chain succession; succession; and co-existence constraints) and with data conditions.

5.2 Result analysis

Some of the learned rules may represent common knowledge to the scientist (such as the pattern that “Dataset Training Loader” is the initial activity and “Dataset Tokenization” activity only occurs after “Data-set Training Loader” activity, meaning that dataset tokenization should always successes data loader). Other rule is that after “Dataset Training Word Removal”, the scientist can either execute “Dataset Training Special Word Removal” or “Dataset Training Stop Word Removal”, *i.e.*, these two activities occur after “Dataset Training Word Removal”, these activities co-exist in the workflow, *i.e.*, the “Dataset Training Special Word Removal” activity can take place before or after “Dataset Training Stop Word Removal”, there is no difference in the expected result.

Thus, in order to meet user requirements, the scientist can follow the rules represented in the combined model, *i.e.*, workflows should be modeled according to the events flow and to the set of parameters kfold>=3.5 $\wedge$.lword< 5 $\wedge$ corpus=politics.products.

The proposed approach was also applied to discover the data-aware declare rules that lead to unsuccessful scenarios. We chose the branch that does not meet user requirements. With the data-aware constraints kfold<1.5 $\wedge$ lword>=2.5 $\wedge$ corpus=politics_products the extracted model is depicted in Figure 18, we can observe that event flow and declare constraints differ when compared to the model found in successful scenarios (Figure 17). The scientist may analyze the rules represented in the combined model and realize that some activities like “DataSetTrainingStopRemoval” are missing.

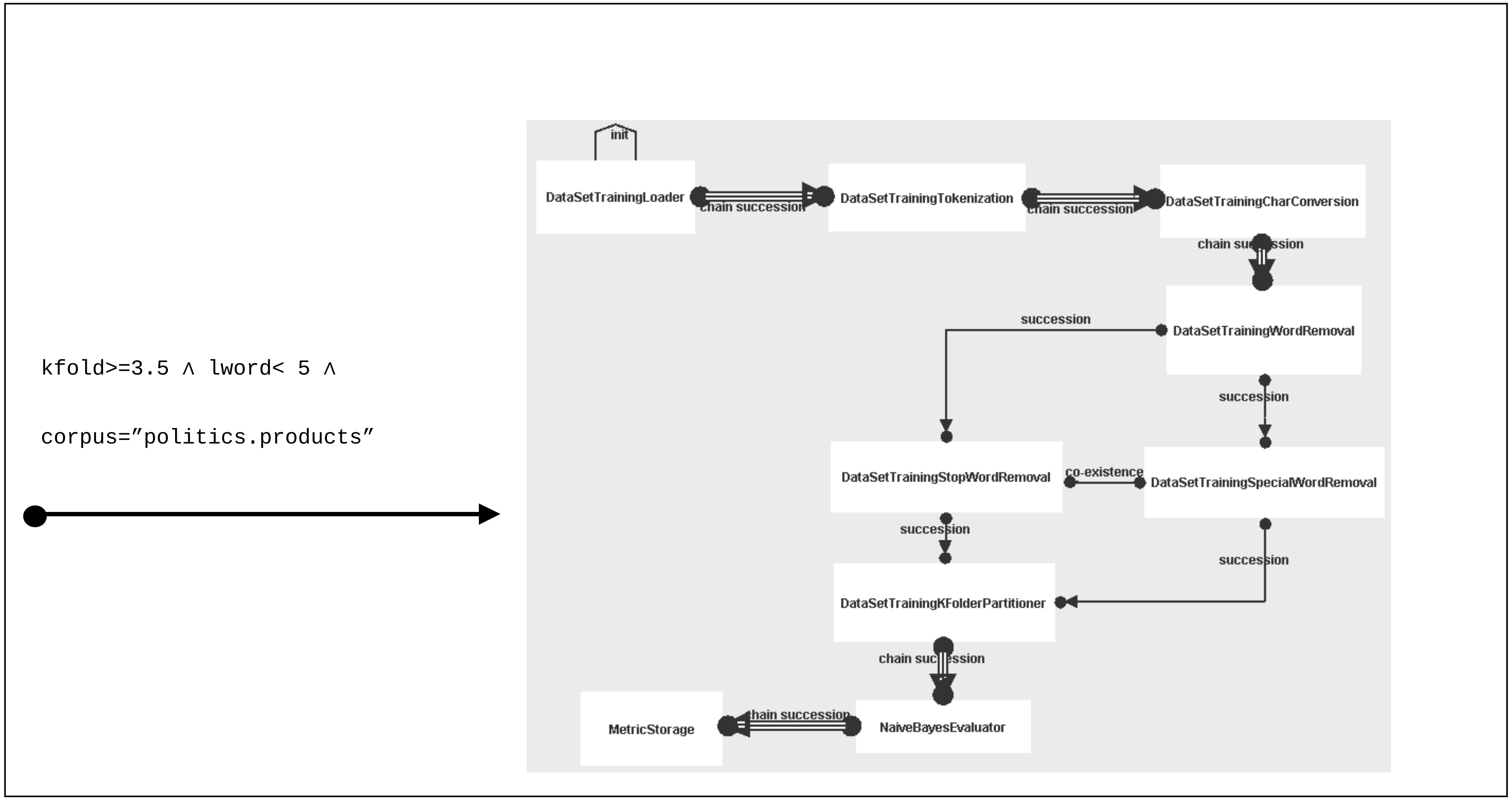


Figure. 17 - Learned Declare Model

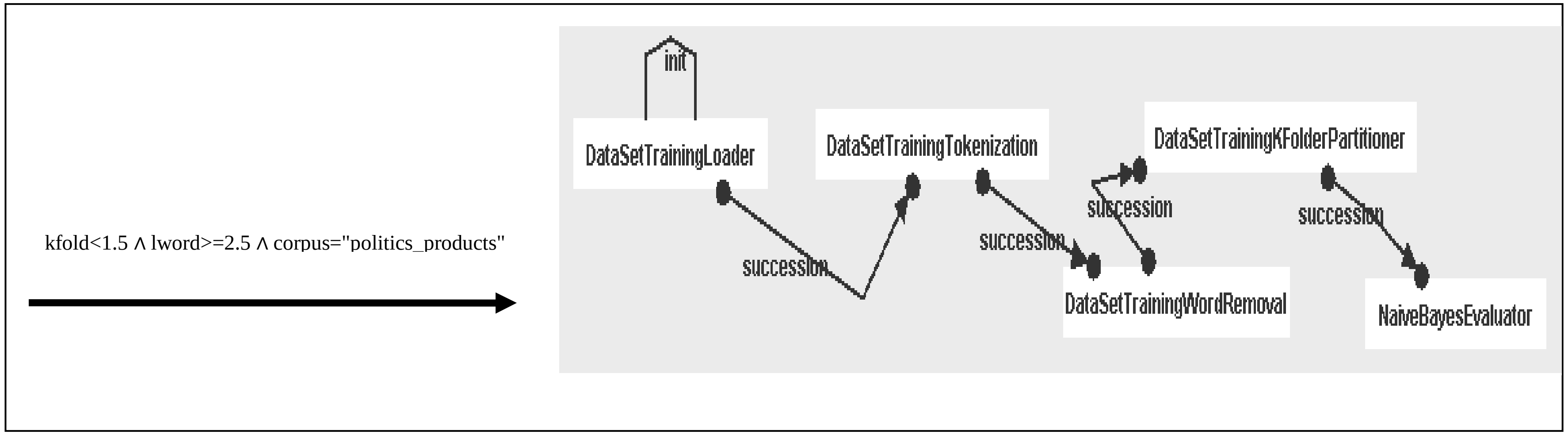

kfold<1.5 ∧ lword>=2.5 ∧ corpus="politics_products"
init
DataSetTrainingLoader
DataSetTrainingTokenization
DataSetTrainingKFolderPartitioner
succession
succession
succession
succession
DataSetTrainingWordRemoval
NaiveBayesEvaluator

Figure 18 - Learned Declare Model (unsuccessful scenarios)

5.3 Evapotranspiration Experiment

In this section, we present an experiment of an evapotranspiration estimation scenario. Evapotranspiration is the sum of water evaporation from the soil with the transpiration from vegetation which returns to the atmosphere in vapor form. The values of evapotranspiration are important in agriculture, especially in soil irrigation activities. Thus, the availability of evapotranspiration values allows a better planning of the water amount to be irrigated and thus minimize resources use and environmental impacts. However, the direct measurement of evapotranspiration is difficult and costly, since it requires facilities and special equipments. Thus, the use of knowledge discovery techniques was used in this scientific experiment (XAVIER *et al.*, 2015).

The experiment was done with weather data and showed that some specific parameter values combinations presented better results when compared to historical data of the estimated evapotranspiration. The data was extracted from Meteorological Database for Education and Research (BDMEP) of National Institute of Meteorology of Brazil (INMET). This database has had historical series of weather data for several locations in Brazil since 2006, collected through measuring stations. These historical series contain data evapotranspiration, which were used as input data for the learning algorithms. This datasets have the variables: wind speed average, high speed wind average, piche evaporation, potential evapotranspiration, total insolation, cloudiness average, total precipitation, medium pressure, maximum temperature average, compensated temperature average, minimum temperature average, relative humidity average. The scientist applied a linear regression technique to measure the degree of precision between the historical values and the values obtained by the parameter combination. The coefficient of determination, also called R^2, is a linear approximation of the general statistical model. R^2 varies between 0 and 1, indicating the percentage of the observed values that is consistent with the learned model (XAVIER *et al.*, 2015).

The workflow has six activities (Table 7); some of them with varying parameters, such as: “dataset” in activity 1, “attribute filter” in activity 2, “class” to be clean in activity 3, and “mining algorithm” in activity 4. The “coefficient of determination” is calculated in correlation compute activity.

The modeling of scientific workflow in SWfMS was part this work. The workflow template was specified and executed through the SciCumulus SWfMS on top of the Amazon EC2 cloud environment, using Linux instances (large server and micro instances). In this parallel environment, the scientist took advantage of the parameter sweeper function of SciCumulus (OLIVEIRA *et al*. 2010), generating more than 15,000 combination executed in 2000 workflow instances. Provenance data was collected during workflow executions and stored in a PostgreSQL relational database.

Table 7 - Evapotranspiration Workflow Activities

Activity no.	Activity Name	Activity Description	Parameter	Parameter Description	Data Domain
1	Dataset Loader	load data of station	dataset	station name to be load	nominal (34 stations of Brazilian state Bahia and Rio de Janeiro)
2	Attribute Filter	remove variables	attr1, attr2	range of attributes that will be removed	attribute1: 1 to 5, 7 to 19 attribute2: 2-6, 8-20
3	Data Clean	clean the variable to be analyzed	class	class to be analyzed	7
4	Data Mining	perform data mining activity	algorithm	algorithm name	bagging, decisionstump, gaussianprocesses, m5p, multilayerperceptron, paceregression, reptree, rbfnetwork
5	Correlation Compute	compute the correlation measure	NA		
6	Metric Storage	store the metric of workflow instance	NA		

The user-requirements for filtering provenance data were set by the scientist as follows: “correlation >0.9” was the chosen metric for analyzing workflow instances. In order to discover data-aware constraints, we applied the J48 algorithm as the classification learning algorithm due this algorithm present a tree more pruned than Random Tree algorithm, from Weka toolbox (HALL *et al.*, 2009). Figure 19 shows the branches in classification tree with parameters that guide the experiment to meet the metric "correlation >0.9".

After applying each constraint, the filtered provenance data was sent as input data to the Declarative Process Mining Tool. In this use scenario, we adopt

DeclareMiner plug-in (MAGGI et al. 2011) in ProM tool as the declarative process mining tool.

Due to simplicity for illustration purposes, the extracted model considered only init, chain succession and co-existence declare constraints. The learned data-aware declarative model presents knowledge combining constraints on events flow and activities parameters. Figure 20 depicts the extracted model (considering only init; chain succession; succession; and co-existence constraints) and with data conditions.

5.4 Result analysis

This learned data-aware declarative model (Figure 20) informs that “DatasetLoader” or “AttributeFilter” are the initial activities, i.e., the past execution of experiments started sometimes with “DatasetLoader” activity or other times with “AttributeFilter” activity. More over, the constraints “*chain succession*” said that the tasks “DatasetLoader”, “AtributeFilter”, “DataClean”, “DataMining”, “Correlation Compute”, “Metric Storage” happen in chain.

The scientist sometimes executed “M5P” activity instead of “DataMining” activity. In “DataMining” activity is possible vary the algorithm like parameter, but in “M5P” the algorithm is not a parameter, actually, this activity implement in code the algorithm m5p as fixed. The constraint *not co-existence(DataMining, M5P)* inform that the activities do not occurs at same workflow instance and using one or other the experiment combined with data-aware constraints met the pre-defined metric "correlation >0.9".

A not expected activity “EvPEstimation” was discovered in provenance data “. This activity is a program not modularized that execute all activity: “DatasetLoader”, “AtributeFilter”, “DataClean”, “DataMining”, “Correlation Compute”, “Metric Storage” in a unique code. The constraint *not co-existence(EvPEstimation, DatasetLoader),* it means that was performed “EvPEstimation” activity or “DatasetLoader” activity.

The data-aware constraints show the combinations of parameters that were performed combined with declarative constraints to have a workflow execution with correlation > 0.9. The rule dataset= rj* ^ attr1<=2 ^ attr2= ^ class = 7 ^ algorithm = “bagging” informs that if you combine discovered declare constraints with any station data of state of Rio de Janeiro; with other parameters: attr1= 1 or att1= 2; any value de attr2 (2-6 or 8-20); class=7; and algorithm= “bagging”, you will the expected result. If you want use station data of state of Bahia, we need choose other parameter combination like • dataset= ^ attr1<=2 ^ attr2= ^ class = 7 ^ algorithm = “paceregression”. Below following the other data-aware constraints discovered.

```
dataset= rj* ^ attr1<=2 ^ attr2=  ^ class = 7 ^ algorithm = “bagging”
dataset=   ^ attr1<=2 ^ attr2= ^ class = 7 ^ algorithm = “paceregression”
dataset= rj*   ^attr1<=2 ^ attr2  ^ class = 7 ^ algorithm = “multilayerperceptron”
dataset=   ^attr1<=2 ^ attr2  ^ class = 7 ^ algorithm = “gaussianprocesses”
dataset= rj*  ^attr1<=2 ^ attr2  ^ class = 7 ^ algorithm = “reptree”
dataset=   ^ attr1<=2 ^ attr2  ^ class = 7 ^ algorithm = “m5p”
dataset=   ^ attr1=3 ^ attr2=4  ^ class = 7 ^ algorithm = “paceregression”
dataset=   ^ (attr1> 3 ^ attr1<=5) ^ attr2=5  ^ class = 7 ^ algorithm = “paceregression”
dataset= rj*  ^ attr1 =9 ^ (attr2>11 ^ attr2<=17)  ^ class = 7 ^ algorithm = “paceregression”
dataset= rj*  ^ attr1=10 ^ (attr2>16 ^ attr2<=17)  ^ class = 7 ^ algorithm = “paceregression”
dataset= rj*  ^ attr1=11  ^ (attr2>11 ^ attr2<=17)  ^ class = 7 ^ algorithm = “paceregression”
dataset= rj*   ^ (attr1>2 ^ attr1<=6) ^ attr2<=17  ^ class = 7 ^ algorithm = “gaussianprocesses”
dataset= rj*   ^ attr1 >8 ^ attr2<=17  ^ class = 7 ^ algorithm = “gaussianprocesses”
dataset= rj*  ^ attr1>16 ^ attr2>17  ^ class = 7 ^ algorithm = “gaussianprocesses”
```

To receive a feedback from a scientist responsible for Evapotranspiration experiment, we elaborate some questions to verify if data-aware declarative model is useful to him for a research team. The data-aware declarative model (figure 20) and a declarative model with same declare constraint was presented to a scientist for he was able to answer the questions. Although, we understand that it will be necessary make a survey with a more comprehensive scientist group, it is possible to verify in this user scenario that the learned data-aware declarative model provides useful information to scientist plan new scenarios of experiments and share previous scientific knowledge. The scientist said that “the data-aware declarative model shows me what scenarios have been assessed, enabling new scenarios to evaluate or validate the scenarios already executed".

The answered questions are listed below. The objective answers were marked with X.

- Question 1: Do you think that the learned data-aware declarative model gives you relevant information?

(X) Strong Agree

( )Agree.

( ) Disagree.

( ) Strong Disagree.

- Question 2: Do you think that the learned data-aware declarative model is more accurate than learned declarative model ?

  ( ) Strong Agree

  (X) Agree

  ( ) Disagree

  ( ) Strong Disagree

- Question 3 - Do you think that the learned data-aware declarative model is important to share knowledge in scientific learning process?

  ( ) Strong Agree

  (X) Agree

  ( ) Disagree

  ( ) Strong Disagree

- Question 4 - Do you think that the learned data-aware declarative model can support you fine-tune your experiment ?

  (X) Strong Agree

  ( ) Agree

  ( ) Disagree

  ( ) Strong Disagree

- Question 5 - Do you think that the learned data-aware declarative model gives you insight to plan new scenarios of experimental ?

  (X) Strong Agree

  ( ) Agree

  ( ) Disagree

  ( ) Strong Disagree

- Question 6- What relevant information or insight the data-aware declarative model gives you? (open question)

  - The data-aware declarative model shows me what scenarios have been assessed, enabling new scenarios to evaluate or validate the scenarios already executed.

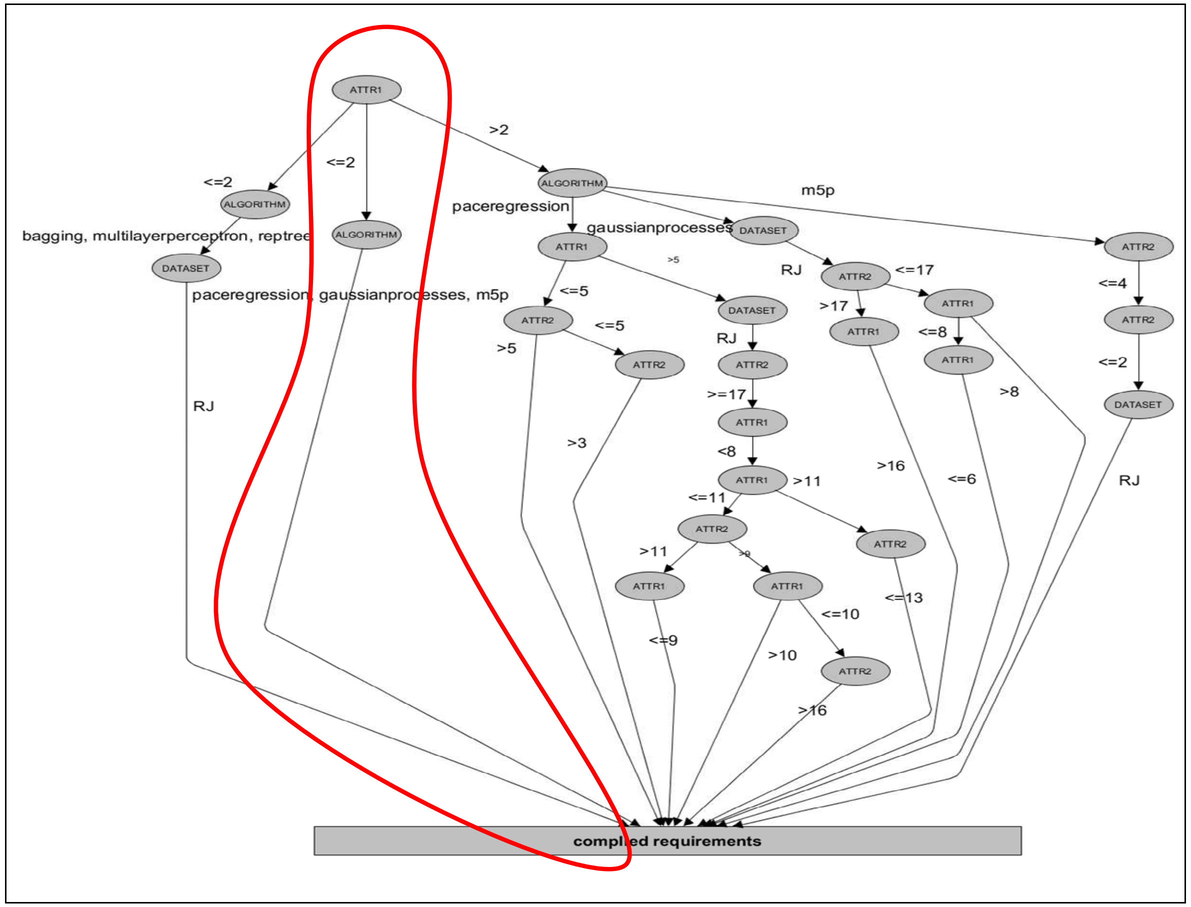


Figure 19 – Decision Tree

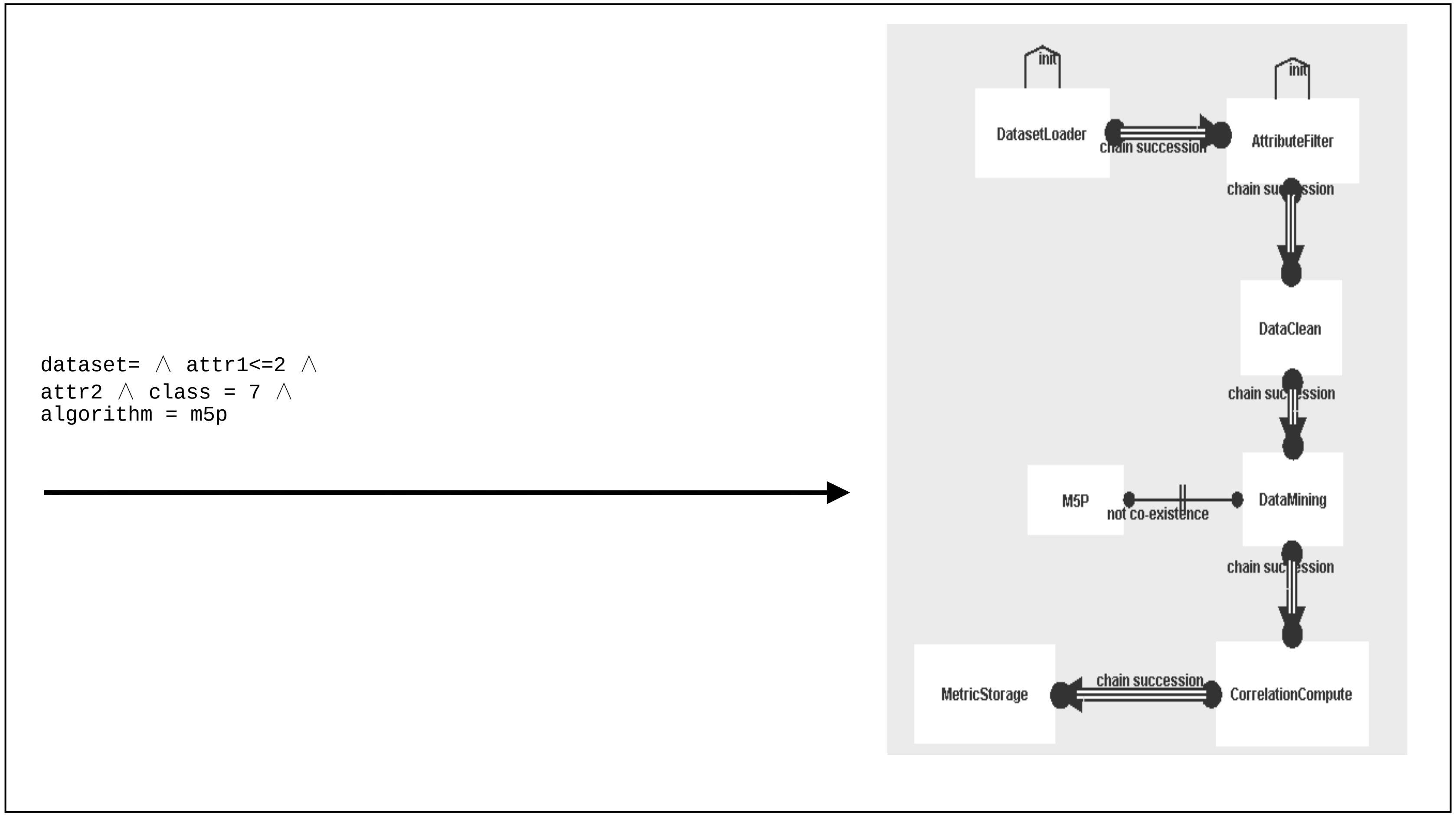


Figure 20 – Learned Data-aware Declarative Model

5.5 Experiments Evaluation

We evaluated the approach with provenance data generated in both use scenarios described in Sections 5.1 and 5.3. The Declare Analysis plug-in (BURATTIN et al., 2012) was applied for analyzing the discovered data-aware declare models resulted from our approach, and to compare it with the traditional declare model (with no data constraints). Both models were compared against the provenance data log, in order to report the metrics. We adopted the metrics: fulfillments ratio and violation ratio (BURATTIN, 2012) to evaluate the experiments.

The extracted log from the first experiment contained 990 workflow instances, and three data attributes (corpus, l-word and k-fold). While, the extracted log from the second experiment contained 2000 workflow instances, and three data attributes (dataset, attr1, attr2, class, mining). Table 8 and 9 show the evaluation metrics (averages of violation ratio and averages of fulfillments ratio) of both experiments for each individual discovered constraint ("init", "chain succession", "succession", "co-existence", "not co-existence"), as well as for all discovered constraints.

Comparing the metrics for both models, we notice that the average fulfillment ratio increased in the data-aware model; this indicates that, in this particular case, the learned data-ware declare model was more accurate. More over, in second experiment, it is possible to verify that without the data-aware conditions the constraint chain succession was not discovered, omitting an important information in learned model.

Table 8 - No. of violation and fulfillment ratio of the wish detection experiment

Constraints	Avg. Violation Ratio	Avg. Fulfillment Ratio	Avg. Violation Ratio	Avg. Fulfillment Ratio
	Traditional declarative model		data-aware declarative model	
init	0.046	0.9540	0.0339	0.9661
chain succession	0.1606	0.8394	0.1218	0.8782
succession	0.1138	0.8862	0.0734	0.9266
co-existence	0.1081	0.8919	0.0366	0.9634
all constraints	0.2105	0.7895	0.1445	0.8555

Table 9 - No. of violation and fulfillment ratio of the evapotranspiration estimation experiment

Constraints	Avg. Violation Ratio	Avg. Fulfillment Ratio	Avg. Violation Ratio	Avg. Fulfillment Ratio
	Traditional declarative model		data-aware declarative model	
init	1	0	0.5473	0.4527
chain succession	-	-	0.1297	0.8703
succession	0.2001	0.7999	0.1292	0.8708
not co-existence	0.2573	0.7427	0.0834	0.9166
all constraints	0.3989	0.6011	0.2936	0.7064

Chapter 6 – Related Works

The approach presented in (MAGGI *et al*., 2013) is related to ours from the perspective that both combine declarative model with data-aware model, this one also through classification algorithm and declare miner. However, firstly, it is discovered a declarative constraints and then they are associated data-attributes. This approach may lead to an inconsistent declarative model, because when it was selected a data attribute, the sub-set is not the same that was used as input for discovering the model. This issue does not happen with our proposal, because the learned data-aware rules are applied to learn a declare model.

While, KNUPLESCH *et al.* (2013) proposes extensions for visual compliance rule based on Compliance Rule Graph (CRG) language to support data, time, and resource perspectives for business processes, but this approach does not combine discovered models.

Another work (MOORE et al., 2013) proposes a declarative language for processing provenance data. While the approach proposed in (BOWERS et al. 2012) infers data dependencies from workflow execution traces based on explicit user-defined rules, they propose a high-level language for expressing dependency rules that are converted in relational queries.

Although the above-mentioned approaches can be used to improve scientific workflows using data provenance, our approach differs since a data-aware declarative model, that represents a workflow instance based on user-requirements, is discovery.

Chapter 7 – Concluding Remarks

Provenance data is an important part of scientific workflows, because it assists scientists on managing, understanding and reproducing their experiments (GIL et al. 2007). However, the amount of provenance data generated from scientific workflow executions grows exponentially through time, becoming infeasible for scientists to manually analyze its content. Thus, mechanisms for extracting and representing knowledge implicit in provenance data are demanding. The use of data mining techniques is a way of automatically discovering useful knowledge from provenance data and presenting it to the scientist in order to facilitate the analysis of his/her scientific experiment. In this research we present an approach that combines two data mining techniques. The first learns data-aware constraints through a decision tree learning algorithm and the second learns declare constraints through ProM declare miner. Therefore, our approach learns a data-aware declarative model in the context of scientific experiments based on use of process mining techniques for learning data-aware declarative models on top of scientific workflows provenance data.

This model provides operational support for improving understandability of the scientific experiment, *e.g.*, a valuable insight for scientist in understanding the main characteristics of his/her data mining experiment that led to successful (or to unsuccessful) workflow scenarios, in planning future executions, *i.e.*, which event flow should be used and which parameter values of their activities. Furthermore, it can be used to conformance checking of semantic-based workflow activities, *e.g.*, if quality metrics for assessing the workflow result were calculated and recorded in a database. Specifically, our approach automatically learns a data-aware declarative model that is more precise with respective to the subset of provenance data it represents. This precision increases more the reliability in information to be used in the scientific learning process.

Our approach was evaluated on a data mining scientific experiment scenario and an evapotranspiration estimation experiment, showing that it is possible to combine the two perspectives of constraints in a unique view.

Some limitations were identified in our research, among them we can mention:

- It is necessary have an enough amount of qualified workflow instances. If not, the learned model may not be more accurate.
- Should have trials with different parameters values for decision tree discovery.
- The parameters of workflow activities cannot be persisted in unstructured data.
- The research team must have familiarity with declarative language.
- Some parts of learning process are error prone.

Time limitations prevented us from fully automating the learning process. Therefore, we propose, as future work, a plug-in implementing in ProM tool. Additionally, more quantitative analysis of the approach to be conducted considering other experiments. Also, an investigation with other declarative languages, such as DCR Graph (Dynamic Condition Response Graphs). Finally, a survey with a comprehensive scientist group will be useful to qualitative analysis the data-aware declarative model in scientific workflow scenarios.

References

ALTINTAS, I., BERKLEY, C., JAEGER, E et al. “Kepler: an extensible system for design and execution of scientific workflows”. In: *Scientific and Statistical Database Management. Proceedings*, 16th International Conference, pp. 423-424, Greece, 2004 .

BOSE, R. J. C., MAGGI, F. M., VAN DER AALST, W. M. “Enhancing declare maps based on event correlations”. In: *Business Process Management.* pp 97-112, Springer Berlin Heidelberg, 2013.

BOWERS, S., MCPHILLIPS, T., LUDÄSCHER, B. “Declarative rules for inferring fine-grained data provenance from scientific workflow execution traces”. In: *Provenance and Annotation of Data and Processes*, pp. 82-96. Springer Berlin Heidelberg , 2012.

BURATTIN, A., MAGGI, F. M., VAN DER AALST, W. M., SPERDUTI, A. “Techniques for a posteriori analysis of declarative processes”. In: *Enterprise Distributed Object Computing Conference (EDOC), 2012 IEEE 16th International* , pp. 41-50, September, China, 2012

CALLAHAN, S. P., FREIRE, J., SANTOS, E., et al. “VisTrails: visualization meets data management”. In: *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, pp. 745-747, Chicago, USA, 2006.

CHENEY, J., CHITICARIU, L., AND TAN, W. C. “Provenance in databases: Why, how, and where”. In: Foundations and Trends in Databases, *Now Publishers Inc*, vol. 1, n. 4, Hanover, USA, 2009.

COSTA, F., SILVA, V., DE OLIVEIRA et al. “Capturing and querying workflow runtime provenance with prov: a practical approach”.. In: *Proceedings of the Joint EDBT/ICDT 2013 Workshops*, pp. 282-289, Genoa, Italy, March, 2013.

CUEVAS-VICENTTÍN, V., DEY, S., KÖHLER, S., RIDDLE, S., LUDÄSCHER, B.: “Scientific work-flows and provenance: Introduction and research opportunities”. In: Datenbank-Spektrum, pp. 193-203, 2012.

DEELMAN, E., CHERVENAK, A. “Data management challenges of data-intensive scientific workflows”. In: *Cluster Computing and the Grid, CCGRID'08, 8th IEEE International Symposium*, pp. 687-692, Piscataway, USA, 2008.

GONÇALVES, J. C. A., BAIÃO, F., SANTORO, F., REVOREDO, K. “Discovering Intentions and Desires within Knowledge Intensive Processes”. In: BPM workshop, Innsbruck, September, 2015.

GIL, Y., DEELMAN, E., ELLISMAN, M et al. “Examining the challenges of scientific workflows”. In: *IEEE Computer*, pp. 26-34..2007.

HILDEBRANDT, T. T.; MUKKAMALA, R. R. “Declarative event-based workflow as distributed dynamic condition response graphs”. In: arXiv preprint arXiv:1110.4161, 2011.

HULL, D., WOLSTENCROFT, K., STEVENS, R. et al. “Taverna: a tool for building and running workflows of services. Nucleic acids research”, 34(suppl 2), W729-W732, 2006.

KNUPLESCH, D., REICHERT, M., LY, L. T., KUMAR, A.,RINDERLE-MA, S. “Visual modeling of business process compliance rules with the support of multiple perspectives”. In: *Conceptual Modeling*, pp. 106-120. Springer Berlin Heidelberg, 2013.

HALL, M., FRANK, E., HOLMES, G. et al. “The WEKA data mining software: an update”. ACM SIGKDD explorations newsletter, 2009.

LIM, C., LU, S., CHEBOTKO, A., FOTOUHI, F. “Prospective and retrospective provenance collection in scientific workflow environments”. In: *Services Computing (SCC), 2010 IEEE International Conference*, pp. 449-456, 2010.

MAGGI, F. M., MOOIJ, A. J.,VAN DER AALST, W. M. “User-guided discovery of declarative process models”. In*: Computational Intelligence and Data Mining (CIDM)*, pp. 192-199, 2011.

MAGGI, F. M., DUMAS, M., GARCÍA-BAÑUELOS, L. MONTALI, M. “Discovering data-aware declarative process models from event logs”. In: *Business Process Management*, pp. 81-96. Springer Berlin, Heidelberg , 2013.

MAGGI, F. M., BOSE, R. J. C., & VAN DER AALST, W. M. “A knowledge-based integrated approach for discovering and repairing declare maps”. In: *Advanced Information Systems Engineering*, pp. 433-448, Springer Berlin, Heidelberg, January, 2013.

MATTOSO, M., WERNER, C., TRAVASSOS, G. H., BRAGANHOLO, V., OGASAWARA, E., OLIVEIRA, D., AND MARTINHO, S. “Towards supporting the life cycle of large scale scientific experiments”. International Journal of Business Process Integration and Management, v. 5, n.1, pp. 79-92, 2010.

MEDEIROS, C. B., SANTANCHÈ, A., MADEIRA, E. et al. “Data Driven Research at LIS: the Laboratory of Information Systems at UNICAMP” In: Journal of Information and Data Management, v2, n2, p93, 2011.

MOORE, S., GEHANI, A., AND SHANKAR, N. “Declaratively Processing Provenance Metadata”. In: *TaPP* , April, 2013.

MOREAU , L. AND MISSIER, P.: “PROV-DM The prov data model”. Retrieve from http://www.w3.org/TR/2013/REC-prov-dm-20130430, 2013.

MURATA, T. “Petri nets: Properties, analysis and applications”. In: Proceedings of the IEEE, pp. 541-580, 1989.

OGASAWARA, E. ; DIAS, J. ; SOUSA, V. ; CHIRIGATI, F. ; OLIVEIRA, D. ; PORTO, F. ; VALDURIEZ, P. ; MATTOSO, M. L. Q. “Chiron: a parallel engine for algebraic scientific workflows”. In: *Concurrency and Computation*, v.25, n.16, pp. 2327-2341, 2013.

OLIVEIRA, D., OGASAWARA, E., BAIÃO, F., MATTOSO, M. “Scicumulus: A lightweight cloud middleware to explore many task computing paradigm in scientific workflows”. In: *Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference*, pp. 378-385, Miami, USA, 2010.

OLIVEIRA, D. 2012, *Uma Abordagem de Apoio à Execução Paralela de Workflows Científicos em Nuvens de Computadores*. PhD Thesis, Federal University of Rio de Janeiro, COPPE/UFRJ, Rio de Janeiro, RJ, Brasil 2012.

PESIC, M.; SCHONENBERG, H., VAN DER AALST, W. P. “Declare: Full support for loosely-structured processes”. In: *Enterprise Distributed Object Computing Conference*, EDOC 2007. 11th IEEE International. IEEE, pp. 287-287, 2007.

PESIC, M.; SCHONENBERG, H. ; VAN DER AALST, W. “Declarative workflow”. In: *Modern Business Process Automation*. Springer Berlin Heidelberg, PP. 175-201, 2010.

SELTMAN, H. J. “Experimental design and analysis”. Retrieve from: http://www.stat.cmu.edu/~hseltman/309/Book/Book.pdf , 2015.

SILVA, M. F.; BAIAO, F. A.; REVOREDO, Kate. “Towards Planning Scientific experiments through Declarative Model Discovery in Provenance Data”. In: e-*Science 2014 IEEE 10th International Conference o. IEEE*, pp. 95-98, 2014.

STOERRLE, H. “A Comparison of (e) EPCs and UML 2 Activity Diagrams”. EPK. Vol. 5, 2006.

TERUEL, M. A., TARDÍO, R., NAVARRO, E et al. “A Goal-Oriented Requirements Approach for Collaborative Bussiness Intelligence”. In *Conceptual Modeling*, Springer International Publishing, pp. 423-430, 2014.

VAN DER AALST, W. M., PESIC, M., AND SCHONENBERG, H.: “Declarative workflows: Balancing between flexibility and support”. Computer Science-Research and Development, pp. 99-113, 2009.

VAN DER AALST, W., ADRIANSYAH, A., DE MEDEIROS, A. K. A. ET AL. “Process mining manifesto”. In: *Business Process Management Workshops*, Springer Berlin Heidelberg, pp. 169-194, 2012.

WESTERGAARD, M.; MAGGI, F. “Declare: A Tool Suite for Declarative Workflow Modeling and Enactment”. In: *BPM (Demos)*, v. 820, 2011.

WHITE, S. A. “Introduction to BPMN. IBM Cooperation”, v 2, 2004.

XAVIER, F., TANAKA, A. K., REVOREDO, K. C. “Aplicação de Descoberta de Conhecimento em Bases de Dados na Estimativa da Evapotranspiração: um Experimento no Estado do Rio de Janeiro”. In: *SBSI*, Goiânia, May, 2015.

ZENG, R., HE, X., LI, J. et al. “A Method to Build and Analyze Scientific Workflows from Provenance through Process Mining”. In: *TaPP*, 2011.

ZUGAL, S., SOFFER, P., HAISJACKL, C., et al. “Investigating expressiveness and understandability of hierarchy in declarative business process models”. In: *Software & Systems Modeling*, pp. 1-23 , 2013.